DOCTOR OF PHILOSOPHY

Text and data mining for information extraction for scientific documents

Muhammad, Bello Aliyu

Award date:
2021

Awarding institution:
Coventry University

Text and Data Mining for Information Extraction from Scientific Documents

By

Bello Aliyu Muhammad

PhD

January 2020

Text and Data Mining for Information Extraction from Scientific Documents

By

Bello Aliyu Muhammad

PhD

January 2020

A Thesis submitted in partial fulfilment of the University's requirements for the Degree of Doctor of Philosophy

Certificate of Ethical Approval

Applicant:

Bello Muhammad

Project Title:

Text and Data Mining to Support Systematic Reviews in Software Engineering.

This is to certify that the above named applicant has completed the Coventry University Ethical Approval process and their project has been confirmed and approved as Low Risk

Date of approval:

06 December 2019

Project Reference Number:

P94003

Declaration

I declare that all work submitted is my work and that any other external materials used within my report has been fully referenced.

Content removed on data protection grounds

Abstract

This research has produced a novel approach for information extraction from scientific research documents by means of text and data mining, including machine learning (ML) and automatic text summarisation (ATS). The scientific research documents (SRDs) consist of an unstructured data which does not have any predefined data model nor organised in any predefined manner. The SRDs are, however, organised in a hierarchical structure commonly known as IMRaD. Extracting the desired information from this structured is both challenging and time-consuming. Automated data extraction is useful in optimising certain administrative processes, review of scientific literature/documents (SRDs) etc. The novel approach developed in this research is centred around the SRDs, i.e. the SRDs are used as the case study to develop the novel approach. Therefore, the approach is most suitable for automatic data (information) extraction from the primary studies (SRDs) during a review process but is scalable to any structured document.

The review of literature is a scientific and rigorous process that aims to integrate the empirical research evidence to answer a given research question. The goal of the review, therefore, is to find all the relevant information on a given research question, how it is covered in the literature and extract it altogether into one piece of evidence that answers the review question using a well-defined set of procedures and guidelines. The most challenging stage in the review process is the information (data) extraction from the SRDs. A review process typically involves hundreds or thousands of SRDs.

Therefore, the process of extracting the desired information from such a vast volume of unstructured data in scientific research documents is labour intensive, error prone and time consuming. Better automated approaches can save reviewers time from between 30% to 70%, and for this reason interest in automated information extraction research is growing. Several methods/approaches have been developed to support the review process, but they all have very little support for the automated information extraction from the primary studies (SRDs).

Information extraction from the SRDs is a difficult/challenging process. It involves the identification and/or extraction of the summary of findings (main result), main topics covered and the methods proposed or used in the documents.

Lack of a unified framework has been identified as the main obstacle in automating the data extraction process from the SRDs. Therefore, a framework is needed to automate or semi automate the process of data extraction from the SRDs.

This research has developed a novel framework (approach) for automated extraction of the relevant data from scientific research publications (SRDs). The framework is based on the canonical structure of the SRDs. The canonical structure is the standard format for the body of scientific documents commonly referred to as IMRAD or IMRaD (Introduction, Methods, Result and Discussion & conclusion). Text and data mining, including machine learning and natural language processing technologies were used to achieve the goal.

First, intelligent models were developed to enable the machines understand the canonical (IMRaD) structure, as machines do not have an implicit understanding of this structure. We analysed, experimented and selected three (3) machine learning (ML) methods for this task, viz: Support Vector Machine (SVM), Logistic Regression and Random Forest. Also, the deep learning Convolutional Neural Network (CNN), was trained. Based on the dataset used in this research, all the above ML methods returned a good accuracy, precision and recall but CNN outperformed them all. This was also enhanced by incorporating a hybrid approach to the machine learning process. The hybrid approach involves the evaluation by the human subject experts.

Second, after the identification of the various sections of the canonical (IMRaD) structure above, the desired section containing the relevant data is delineated for further processing, for example the *'Result'* section to extract the data (findings) from the document. To extract the relevant data, the text within the section is then automatically summarised. The research identified and evaluated the appropriate automatic text summarisation (ATS) approaches and methods. The extractive summarisation approach was used. Four (4) extractive ATS methods were selected and used: Frequency-based (TF-IDF score), Graph-based (LexRank and TextRank) and Cluster-based methods. Using the ROUGE standard for evaluating ATS, the TextRank (Graph-based) method achieved the best performance with an overwhelming recall of 81%, and precision of 65%, as per the dataset (text) used in this research.

Acknowledgement

Glory be to God, the Lord of the universe. May His peace and blessings be upon the noblest of mankind, his household and all those who follow his guidance until the end of time, amin.

My deepest appreciation goes to the sponsor of this PhD programme, the Petroleum Technology Development Fund (PTDF), Nigeria, without whom this doctoral research would not have been possible. They sponsored the programme 100% despite their tight budget. Similar appreciation also goes to my supervisory team who worked tirelessly from the beginning to the end of the programme: Dr. Rahat Iqbal the director of studies (DOS), Prof. Anne James (second supervisor) and Dianbasi Nkantah (third supervisor) have provided the needed supervision for a PhD research. They provided the constructive criticism, support and guidance needed. I am highly indebted to Professor Anne James, who was initially my director of studies (D.O.S) but then left the Coventry University towards the end of my first year. She, however, stayed in the supervisory team providing all the necessary supervision, including face to face meeting with me at our offices in Coventry University.

Similarly, my appreciation goes to my employer in Nigeria, the Usmanu Danfodiyo Univesity Sokoto, who granted me the study leave that enabled me to pursue the Programme on full time basis. My special thanks goes to Professor Aminu Mohammed, Professor of Computer Science, Usmanu Danfodiyo Univesity Sokoto, for all the support, prayers and administrative processes he undertook on my behalf. I say to you Jazakallah Khair.

I will also not forget the prayers and guidance of my lifetime guardian, Alh. Bello Muhammad Chanchangi, (Sarkin Samarin Gusau). He has been instrumental in all the good things that happened in my life. May Allah (SWT) reward you with Jannatul Firdaus, amin.

The support of my family will not go unappreciated. To my father, Alh. Muhammadu and my late mother Hajiya Saadatu (May Allah have mercy on her and grant her Jannah, amin), thank you all. To my wife Habiba Abdullahi, my daughter Salma and my son Abdullahi, I love you all and thank you for everything. Thanks to the support and prayers of Dr Abdullahi Ahmed, Dr Bilyamin Romo, Dr. Aliyu Musa Imam all from Coventry University.

I also appreciate my friends and colleagues here in Coventry University, with whom we shared exciting moments and memories: Yahaya I. Shehu, Aminu Bature Shinkafi, Habib Gajam etc.

Finally, I am forever grateful to all those who contributed in one way or another, individually or collectively, towards the success of this research programme but have not been mentioned here. I thank you all.

Table of Contents

List Figures

List of Tables

List of Glossary

SRD – Scientific Research Document

SLR – Systematic Literature Review

CRD – Centre for Review Dissemination

DARE – Database for Abstract of Review of Effects

EBSE - Evidence-Based Software Engineering

SR – Systematic Review

NLP – Natural Language Processing

EPPI - Evidence for Policy and Practice Information and Co-ordinating

SESRA - Software Engineering Systematic Review Automation

PICO - Population, Intervention, Comparison and Outcome

SCAS – Score Citation Automatic Strategy

VTM – Visual Text Mining.

NCD - Normalised Compression Distance

ML – Machine Learning

AI - Artificial Intelligence

SVM - Support Vector Machine

OVA - One-Versus-All

RNN - Recurrent Neural Network

CNN – Convolutional Neural Network

TF-IDF - Term frequency-inverse document frequency

HITS - Hyperlink-Induced Topic Search

NLTK – Natural Language Processing Tool Kit

TP = True Positive

FP = False Positive

FN = False Negative

TN = True Negative

1 Introduction

This chapter presents the background to the research, the use of text and data mining technology for information extraction from the scientific research documents. It begins by putting the background into context as well as the motivation for the research. The chapter also highlights the research problem, the aim and objectives as well as briefly presenting the methodology followed to achieve the stated objectives. The research novelty, i.e. the contribution to the overall body of knowledge, is also spelt out in this chapter. The publications (research outputs) that emanated from the research are also detailed. Finally, the thesis structure, i.e. a summary depicting the chapters with their corresponding contents, is also contained in this chapter.

1.1 Background

1.1.1 Text/Data Mining, Scientific Research Publications (SRPs) and The Information Extraction

Text mining, also referred to as data mining, is an artificial intelligence (AI) technology that involves the use of natural language processing (NLP) to extracts a high-valued information from free (unstructured) text in documents into a format suitable for analysis, training machine learning algorithms etc (Dörre 1999). Due to the exponential growth in the amount of data available today (in both quantity and relevance), the need for robust and scalable text and data mining approaches capable of handling such vast volume of data also increases. Technologies such as machine learning, automatic text summarisation and natural language processing are promising technologies for any text and data mining projects (Muhammad 2019).

Thousands of gigabytes of data are generated on a daily basis from several online and offline sources such as digital databases, social media platforms, news outlets, emails etc. (Marr, 2018). This leads to data deluge. About 80% of the data is unstructured (Beal, 2019). Unstructured data does not have any predefined data model nor is organised in any predefined manner. The scientific research publication (SRDs) are typical example of scientific documents containing unstructured data. The documents, however, are reported in a structure known as IMRaD. Extraction of relevant information from SRDs is useful for tasks such as review of literature.

During a review process, useful information must be extracted from the pool of these free (unstructured) text (SRDs) in order to answer the review question. Processing such vast volume of unstructured data from structured documents requires a robust approach/technique in addition to leveraging the existing technology (Muhammad *et al.,* 2019). Other examples of structured documents are financial documents (Rimchala, 2019). The scale of the research problem is fully discussed in section 1.3.

In this research therefore, a novel approach suitable for information extraction from the SRDs was developed. Although the novel approach in this research is centred around the SRDs, it is, however, scalable to any structured document containing free text. The focus here, however, is on the SRDS typically used for systematic literature review purposes. The motivation for this is contained in section 1.4.

An overview of the review process, which is typically the target of this novel approach is given in section 1.1.2 below. Text and data mining including machine learning, ATS and NLP techniques will be used to obtain the results.

1.1.2 Systematic Literature Review (SLR)

The review of the available literature is the foundation of any scientific research project. This involves reviewing the current state of the science from the available literature to answer a research question as well as to identify a knowledge gap where a new investigation can begin (Budgen and Brereton, 2006), which is driven by the demand for evidence-based practice (Medina and Pailaquilén, 2010). This review process is either systematic or traditional. The traditional or review is not guided by any clear or well-defined, systematic procedures for ensuring that the literature is surveyed objectively (Budgen and Brereton, 2006), thus giving way for a possible bias in the outcome or conclusion of the study (Haddaway and Pullin, 2014). The main purpose of a narrative review is to build an argument on the current state of the science. This involves some bias or cherry-picking of the pieces of literature. The reviewers personally decide which article to keep or throw away using a simple rule: results that support the rationale of the study are included (opinion driven synthesis studies), while the studies that do not are thrown out (Briner and Denyer, 2012; Hakemzadeh, 2012). This lack of thorough and scientific rigour makes the traditional review of little scientific value (Keele, 2007; Molleri and Benitti, 2015).

The systematic literature review (SLR) is a scientific and rigorous data mining process that aims to integrate the empirical research evidence to create a generalisation. It involves the retrieval of the relevant evidence (SRPs), screening the evidence, extracting the relevant data needed to synthesise the evidence related to the research question in a way that is unbiased and (to a degree) repeatable" (Kitchenham, 2004). In performing the systematic review, therefore, several studies from multiple sources on a given research question are considered in order to derive an objective summary of the research evidence concerning that topic of interest (Kitchenham, Dyba and Jorgensen, 2004).

Since SLR takes several primary studies, combines them and produces one overview, the reviews are only as good as the studies (primary sources) from which they are created. This means that the bias in the primary studies cannot be fixed by a systematic review (secondary study) (Zhou *et al.*, 2015*)*.

The goal of the systematic review is to find all information on a given research question, how it is covered in the literature (SRPs) and pull it all together into one palatable piece of evidence for digestion, and to be able to answer the question using a well-defined set of procedures and guidelines (Mulrow, 1994). In this way, research gaps could be identified in order to suggest new areas of further investigation (Tranfield, Denyer and Smart, 2003).

SLR was first developed in biomedicine where it was used by clinicians to keep up to date with the current medical trials since medical experts' opinions were not reliable compared to conclusions from the scientific experiment (Mulrow, 1994), hence the need for SLR. They are now used in many areas and across different disciplines including software engineering (Kitchenham, 2004), social science (Hakemzadeh, 2012), and sciences and education (Cant and Cooper, 2010).

1.1.2.1 Phases and Stages of the systematic review

A systematic review involves several discrete sets of activities performed in phases that follow a defined strategy. The output of each phase feeds into the next until the entire process is complete (Kitchenham, 2004). Based on the systematic review guideline proposed by Kitchenham (2004), there are three (3) main phases of SLR: planning, conducting and reporting (Briner and Denyer, 2012; Kitchenham, 2004; and Zhou *et al.,* 2015).

The review process begins with the identification of an answerable research question. The research question normally determines the search string and the kinds of literature to be identified (Brereton *et al.,* 2007). This may be followed by a short description of the research problem or information need, followed by the systematic review phases. Some of the activities in these phases are mandatory while others, such as commissioning the review, evaluation of the review protocol and evaluation of the reports are considered optional. This is because a review is only commissioned if it is undertaken for commercial purposes. The evaluation of the review protocol and report will significantly depend on the quality assurance protocols put in place by the review team. These phases and their sub-phases are described below.

Phase 1: Planning Phase

The planning phase kick-starts the entire SLR process, detailing how it should be performed. The planning phase consists of the following activities.

- **Activity 1:** Identification and justification of the need for the review. At this stage, an exhaustive search for any existing review on the topic is performed, to avoid re-inventing the wheel.
- **Activity 2:** Development of the review protocol. The review protocol is a detailed plan to be followed. It guides the selection of the primary studies as well as their quality assessment. It also details the allocation of reviewers to the various activities of the process. A pre-defined protocol is necessary to reduce the possibility of a reviewer bias.
- **Activity 3:** Validation of the review protocol against possible bias/errors. The validation can be done by experienced reviewers or by non-experts through piloting the research protocol for any possible error in the data collection and aggregation procedures (Keele, 2007). The procedure for the evaluation of the review protocol must be agreed by the researchers. This must also anticipate the possibility of changing the review question, synthesis methods and data extraction forms.

Phase 2: Conducting the review

Once the reviewers have agreed on the plan, the review begins. However, the entire process is iterative, i.e. the planning phase can be returned to, anytime, should the need arise. Activities in this phase include:

- **Activity 1:** Identification of primary studies: All relevant studies should be identified as much as possible. Identification and retrieval should be done using an unbiased search strategy. Software engineering search engines are not designed to support systematic reviews, as such the researchers perform source-dependent searches (Kitchenham, 2004). Using an efficient search string, multiple electronic sources need to be searched, including non-published studies. This is because no single source finds all the relevant papers (Kitchenham, 2004). Possible sources of primary studies in software engineering include IEEExplore, ACM Digital library, Google scholar, Citeseer library, ScienceDirect, EI Compendex, Springer link, SCOPUS etc. (Brereton *et al.*, 2007).
- **Activity 2:** Selection of primary studies: After retrieving the studies, the reviewers then select the most relevant ones that address the research topic. This is done using the study selection criteria defined in the review protocol intended to reduce the likelihood of bias. An inclusion/exclusion criterion should be used to select the studies to ensure that they can be reliably interpreted to classify and exclude the studies

correctly based on titles and abstract. First, the titles and abstracts are read to find the relevant ones. These are then accepted and read in full. This two-way process is necessary because the abstracts in the software engineering domain are often poorly crafted (Brereton *et al.*, 2007); hence, the need to read the full text of the paper. It is important that this process is executed by two (2) reviewers, allowing any disagreement on the included/excluded papers to be resolved.

- **Activity 3:** Study quality assessment: This is the process of weighing the relevance or importance of the included primary studies in addition to the inclusion/exclusion criteria. There is no agreed definition of quality but there are guidelines (Kitchenham, 2004). Some of the guidelines are provided at a Centre for Review Dissemination (CRD) Database for Abstract of Review of Effects (DARE) (Matters, 2002). The guidelines suggest that the included study should minimise bias and maximise internal and external validity.
- **Activity 4:** Data extraction: This involves the extraction of the relevant data from the included primary studies that correctly answers the research question. The data is usually extracted into forms. To reduce the possibility of bias, the data extraction forms should be defined during the protocol definition. The content of the data extraction forms includes the names of the reviewers, the date of the extraction, title, author, journal, publication details, etc. The findings from these included studies are then used for the data synthesis to answer the research questions.
- **Activity 5:** Data synthesis: This is the presentation of the summary of the included primary studies. Synthesis can be quantitative or qualitative (descriptive). Quantitative synthesis involves using statistical techniques to present the results and is referred to as a meta-analysis. Data synthesis can also be specified in the review protocol.

Phase 3: Reporting the review

This is the final phase of the review process. It involves writing the results of the review, summarising the above first two (2) phases for dissemination to the interested parties. In this stage, the following activities are carried out:

- **Specifying the dissemination mechanism**: For academics, dissemination is about reporting the result of the review in a journal article. However, the practitioners have intended beneficiaries of the review, hence other forms of dissemination must be provided such as posters, press release, summary leaflets etc.

- **Formatting the report:** The report must be formatted in such a way that the rigour of the exercise can be assessed by the readers or the targeted audience. For reviews reported in the journals which have a size restriction, the technical report containing the details must be pointed to.
- **Evaluation of the report:** The reviews reported in the journals are usually peer-reviewed but technical reports are not subjected to any form of independent assessment.

1.2 Evidence-Based Software Engineering (EBSE)

The SRPs are very similar across all disciplines. However, this research used the SRPS from the software engineering (SE) domain. Notwithstanding, SE SRPs follow a stricter hierarchical structure in reporting (Kitchenham, 2004). Hence, approaches that works fine with them could easily be scaled to SRPs in other domains (Muhammad *et al.,* 2018). The choice of software engineering SRPs is based on the trend of the demand for EBSE, which seeks to close the gap between the research and practice.

EBSE aims "to improve decision making related to software development and maintenance by integrating current best evidence from research with practical experience and human values" (Dyba, Kitchenham and Jorgensen, 2005). For example. the IT companies frequently make decisions on the choice of technology for the implementation of various projects. To make a good choice, therefore, practitioners embark on EBSE as a mechanism to support and improve their technology adoption decision. This requires information extraction from several SRPs to answer the research questions that guide the decision-making process.

Similarly, SE research does not involve taking randomised control trials (RCTs) which involve trials of treatment under its actual use conditions. Experiments are performed in the laboratory which is not considered to provide compelling evidence. "This implies that SE shouldn't rely solely on laboratory experiments and should attempt to gather evidence from industrial projects, using observation studies, case studies, surveys, and field experiments. These empirical techniques do not have the scientific rigour associated with the formal randomised experiments, but they do avoid the limited relevance of small-scale, artificial SE experiments" (Dyba, Kitchenham and Jorgensen, 2005). Thus, the major issue for software engineering study is whether small-scale experiments are considered the equivalent of laboratory experiments evaluated at the lowest level of evidence (Kitchenham, 2004). Therefore, experiments performed in academic settings cannot be equated to RCTs in medicine; hence, the need for the EBSE.

Budgen *et al.* (2006) compared the research practices of software engineering with other domains. They concluded that software engineering is significantly different from the medical fields. However, it is closer to social sciences, hence EBSE guidelines were visited to incorporate the ideas from social sciences (Petticrew and Roberts, 2008).

EBSE involves the following steps:

1. Convert the given problem or information need into an answerable question.
2. Conduct a thorough search for the best available evidence that answers the question.
3. Critically appraise the evidence for validity, impact and applicability.
4. Integrate the software engineering expertise with the appraised evidence.
5. Evaluate performance and identify the best ways to improve it.

It is worthy of note that there is no special role played by the SE SRPs used for the model training. SRPs from other disciplines could as well be used and would produce similar result. This only advantage is that SE SRPs follow strict structure which led the model to be more robust.

1.2.1 Characteristics of Systematic Reviews in Software Engineering

The general overview of systematic reviews, including the various stages of the process has been outlined in section 1.1. Although all the respective stages are similar in all the domains (in which SLR is applied), SE has some peculiar differences in the way the guidelines are adopted and implemented (Kitchenham, *et al.,* 2009). Software engineering involves participants who take an active role in the research (for example, programming), unlike in biomedicine where subjects take some form of treatments (Kitchenham, *et al.,* 2009). The difference is that the outcome of the research may be influenced by the nature of its participants through their expertise or experience. Moreover, the reporting standard for software engineering papers is often poor (Brereton *et al.*, 2007; Kitchenham, *et al.*, 2009). It may be that many primary studies ignore the likelihood that the studies may be used for systematic review in the future (Kitchenham, *et al.*, 2009). Primary studies in software engineering also lack statistical power (Kitchenham *et al*., 2009), because studies usually require specialist skills and knowledge, which makes participant recruitment difficult. Many studies in software engineering, therefore, fall short of the number of participants required to generate (what is generally regarded as) an acceptable level of statistical power (Dyba *et al*., 2005). This again limits the strength of synthesis, which can be achieved in a systematic review and makes performing meta-analysis particularly challenging.

1.3 Research Problem

In this age of big data, characterised by data deluge, the analysis and processing of the huge volume of unstructured documents, such as the SRDs, requires robust and sophisticated methods and techniques. Until now, the tools we use to organise data are incapable of dealing with such variety and volume (Debnath, 2020). How then do we deal with the onslaught of the overwhelming data? Effective tools and techniques which would improve the process are of high importance in the management and analysis of this data.

The SLR activity is characterised by an overwhelming and unstructured data (mainly consisting of SRDs), sometimes running into thousands. Hence, the process is labour intensive, error prone and time-consuming. Search engines and on-line bibliographic resources are conventionally used to locate the relevant literature (primary studies) using key word search only. After that stage, however, there is little automated help. The reviewers manually select, assess and synthesise the evidence (data) from the bulk of the included papers into the review. Better computer-assisted support can save in researcher time from between 30% to 70% (O'Mara-Eves *et al.,* 2015), and for this reason, research interest in the automated extraction/mining of relevant data/information from the pool of unstructured data (documents), such as SRDs has gained momentum.

The automation technology itself is not a problem. In fact, some automation/support tools have been developed to support the process including special-purpose and general-purpose (Marshall, 2016). However, the missing piece is the non-availability of a suitable approach/framework to extract the relevant data from the pool of unstructured documents (the SRDs).

Extracting the desired data from SRDs is a challenging task (Jaspers, De Troyer and Aerts, 2018; Marshall, 2016). First, the documents are represented in several different ways (Majumder *et al.,* 2020), hence an approach that accommodates these different ways is needed. Second, the technology has not fully been utilised to automate this most important but challenging activity (data extraction). Extracting the relevant information from the SRDs involves the identification and extraction of relevant but summarised information from the documents. It also includes identifying the main topics covered as well as the methods proposed or used in the studies (Kitchenham and Brereton, 2013).

Jonnalagadda *et al.* (2015) identified the lack of a unified framework as the main obstacle in automated data extraction from SRDs. Any attempt to automate or semi automate the automated data extraction from these documents must consider using a befitting framework.

Active machine learning, where the learning algorithm itself attempts to select the most informative data for training, has been identified as a promising approach to reduce the workload by automating some of the screening decisions, but more evaluation is necessary (Hashimoto *et al.*, 2016). This will ensure that the technology is fully explored to provide the needed support or automation, wherever and whenever necessary. Some research projects have applied text mining to support specialised review tasks (Chang *et al.*, 2016; Millard, Flach and Higgins, 2015); however, more studies are needed to discover the generalisability of the approaches.

Most scientists use computational and automated models with excessive experiments and simulations to run analysis as well as evaluate the performance of the methods without the involvement of human actors for verification. This research would involve the human participants and subject experts; and explore the role they could play to improve the efficiency of machine learning model development and performance.

The extraction of information for summarisation also gives rise to research challenges. Summarisation can be achieved by using extraction where the most relevant sentences are found and taken to be post-processed to form the result. The challenge will be in creating an algorithm that can find the most relevant sentences. Various mining techniques will be explored for applicability and performance.

1.4 Research Motivation and Purpose

Data is the new oil of the economy. It must, however, be processed efficiently for it to have value (Humby 2006). The use of effective text and data mining techniques is critical in handling the exponential growth of data today. The review of available literature is the foundational requirement for most research projects. Scientific research projects involve the processing of thousands of documents (SRDs). In this age of big data, it is interesting to explore how much of this task can be supported and enhanced using text and data mining techniques, especially for data/information extraction from SRDs (Bosco, Uggerslev and Steel, 2014; Gandomi and Haider, 2015). As described in subsection 1.2.1, SLR typically involves the following five (5) steps: Searching the literature using online resources and keywords; screening the papers for those that seem relevant based on title and abstract; reading the selected papers; summarising the most relevant literature to identify the needed answers as well as the commonalities and differences between them; and lastly identify research gaps where new knowledge and research would be helpful. Most research so far has concentrated on the steps (1) and (2) above. This

project will investigate what support can be engineered through text and data mining for steps (3), (4) and (5). Specifically, stages 3 and 4 are where the main research challenges lie, and these are the stages on which our attention was focussed. The challenge is that machines do not have a true understanding and so the artificial intelligence procedures would be created to enable the system to "understand" the material in order to find, analyse and extract the relevant contents required. Text and data mining including machine learning, and deep learning were utilised for this purpose.

The volume of data produced daily has skyrocketed, running into hundreds of gigabytes (Vuletta, 2020), and with the growth of the Internet of Things (IoT) this pace is only accelerating (Lackey, 2019), with over 70% of this data in unstructured format (Wall, 2014; King, 2019). This data comes from several sources, such as social media platforms, news feeds, CMR platforms, emails, digital libraries (databases) etc. For example, in biomedicine alone, more than 1 million papers are dropped into the PubMed database every year. That is about 2 papers per minute (Landhuis, 2016). In fact, a lot of this data becomes stale after just 90 days. Therefore, faster and more efficient approaches are needed to process this vast volume of data to derive value. Hence, the interest in searching for tools and techniques to effectively process this overwhelming growing volume of data. This has motivated the proposed research work.

1.5 Research Questions

This research contributes to the ongoing research efforts to developing approaches that enable the automated extraction of data from the SRDs particularly to support the SLR process. This research focused on the software engineering domain. We took a broad view of what constitutes the software engineering domain including software design, development and innovation. The research sought to answer the following research questions.

- Cruzes *et al.* (2007) put forward a very important research question: "*automated data extraction from SRDs for review purpose, is it possible?"* This research question remains just as valid today as it was in 2007 (Jaspers, De Troyer and Aerts, 2018).
- What is the unified approach (framework) to adopt for automated data extraction from the SRDS?
- To what extent can text and data mining technology be applied and developed to discover information from SRDs relevant to a literature review?

To answer the above research questions, a thorough and in-depth analysis of the relevant concepts and technologies was carried out in addition to the appreciation of the works done so far in the area. Also, a robust solution has been proposed by this research.

1.6 Aim and Objectives

The aim of this research is to develop a unified framework for automated extraction of relevant data from the SRDs. This involves using the data mining technologies to help identify and extract the findings (results), methods and conclusions from a published SRD for a given topic and summarise the content.

The objectives of the research are as follows:

1. Carry out an extensive literature review on information extraction, SLR process, text and data mining technologies, natural language processing (NLP); machine learning (including deep learning) and summarisation techniques. In the end, identify the research gap.
2. Develop a canonical model representation (IMRaD structure) of the scientific research documents using the software engineering documents (dataset).
3. Develop machine learning models to recognise (understand) the canonical model (IMRaD structure) developed in 2 above.
4. Exploit a novel hybrid approach to machine learning for prediction. Combine the machine learning models with human expertise to improve the performance of the model.
5. Appraise and select the appropriate summarisation techniques; and apply them to summarise the relevant contents identified automatically according to the canonical model in 3 above.
6. Compare and evaluate the methods in (5) and choose the appropriate for the task.
7. Draw conclusions from the project.

1.7 Original Research Contribution (Novelty)

This is a relatively new area and, so far, research efforts in automated information extraction from SRDs (for review purposes) have concentrated on classifying whether a document is relevant to a review (O'Mara-Eves *et al.*, 2015). This research goes some steps further to extract the most relevant content according to various categories from the SRDs and in addition performs summarisation. Most text mining systems used in a SLR to date have used shallow seed information such as bag of words. This research uses a more advanced natural language processing techniques (comprising of machine learning and automatic text summarisation) to extract and summarise the relevant data.

Th**e** research investigates the use of technology to address the challenge of finding a suitable approach/framework for automated data extraction from SRDs. As highlighted in the problem statement, the automated data extraction is not feasible yet, mainly because there is currently no suitable framework. A unified framework specifically for that purpose has been developed. To the best of the researcher's knowledge, it is the first of its kind.

This research has developed a novel approach (framework) based on the canonical model of the structure of the SRDs which is the first of its kind for use in the natural language processing of SRDs and other related documents. The canonical model is but a representation of the various parts/sections of the SRDs, through which the findings are reported. The machine learning models developed by this research, which can identify and extract the respective contents (sections) from the SRDs, are also of novel note.

This research has also advanced a new idea for modern ML and hybrid AI-NLP research that involves human actors for verification. Most scientists use computational and automated models with excessive experiments and simulations to run analysis as well as evaluate the performance of the methods without the involvement of human actors for verification. Involving the human actors would be useful in improving the prediction efficiency of ML, hence the need for this hybrid approach. This is because the human judgement would help to build better models with better predictive power by leveraging the input (expertise and experience) from the human actors (experts). The human experts annotated the dataset for training and participated in evaluating the results (models) after the training.

Based on the dataset used in this research, the artificial neural network (convolutional) performed better than SVM, logistic regression and random forest in natural language processing (NLP) particularly when the data has many features in different forms. Similarly,

text rank summarisation method is the most suitable for ATS according to the dataset used in this research.

1.8 Thesis Outline

The brief description of the rest of the chapters (excluding chapter one) is given below.

Chapter Two: Literature review

In this chapter, previous works on information extraction from SRDs on unstructured data were reviewed. Including the technologies such as the natural language processing (NLP), text and data mining, the machine learning algorithms (including the deep learning) and automatic summarisation techniques (ATS) are also comprehensibly explored in the chapter. The research gap was also identified

Chapter three: Methodology and experimental design

This chapter describes, in full, the methodology followed to achieve the various objectives stated previously (in subsection 1.5). It also describes the general/overall approach taken to achieve the results including the data collection, the research phases, the experimental design and the overall system integration.

Chapter Four: The canonical model development

Following the details in chapter three, chapter four details the canonical model development. It includes the algorithm design for the canonical model, the data preparation, statistical analysis and the finally, the model development.

Chapter Five: Machine learning development

This chapter covers the machine learning model development which enables the system to understand the canonical model reported in chapter four. It involves experimentation with the various machine learning models including the deep learning models.

Chapter Six: Convolutional Neural Network (CNN)

This chapter covers the application of the convolutional neural network (CNN) to the research task. The CNN is more sophisticated and more robust than the traditional machine learning methods. It resulted in improved results.

Chapter Seven: Summarisation

This chapter contains the details of the summarisation task. It discusses and trials some different approaches to summarisation and evaluates the outcomes using the ROGUE tool.

Chapter Eight: Evaluation

To assess whether the research meets its designed objectives, a system evaluation was carried out by potential users. This evaluation is reported in chapter eight.

Chapter Nine: Conclusion Recommendation and Future Work

The general conclusion of the overall research, highlighting the achieved objectives and future impact of the research, is the contents of chapter nine, the final chapter of the thesis.

1.9 Research Outputs

The following subsections present the research output published/presented.

1.9.1 Publications arising from the Research

1. Aliyu, M. B., Iqbal, R. and James A. (2018). Iqbal, R. and James, A. (2018, October). The Canonical Model of Structure for Data Extraction in Systematic Reviews of Scientific Research Articles. In *2018 Fifth International Conference on Social Networks Analysis, Management and Security (SNAMS)* (pp. 264-271). IEEE.
2. Muhammad, B. A., Iqbal, R., James, A. and Nkantah, D. (2019, November). Convolutional Neural Network for Core Sections Identification in Scientific Research Publications. In *International Conference on Intelligent Data Engineering and Automated Learning* (pp. 265-273). Springer, Cham.
3. Muhammad, B.A., Iqbal, R., James, A. and Nkantah, D., (2020, September). Comparative Performance of Machine Learning Methods for Text Classification. In *2020 International Conference on Computing and Information Technology (ICCIT-1441)* (pp. 1-5). IEEE.
4. Muhammad, B.A., Iqbal, R., James, A. and Nkantah, D., (2020) SED: An Algorithm for Section and Subsection Heading Identification from Unstructured Text Documents. International Journal Computer Science Issue. Vol. 117, issue 6.

1.9.2 Posters arising from the Research

1. Muhammad, B.A., Iqbal, R. and James A. (2018). Automatic Data Extraction in Systematic review. A Poster presented at the First Doctoral Capability and Development Conference, Doctoral College, Coventry University, Tuesday 26th April 2018, at the Elm Bank Doctoral School Coventry University, UK.

1.9.3 Presentations arising from the Research

1. Muhammad, B.A (2017). Use of Boolean search String in Information Retrieval. A seminar paper presented at the faculty seminar, School of Engineering, Environment and Computing, Coventry University. *Writing for Computer science and Engineering*, seminar. Tuesday 10th October 2017 at Technology Park.
2. Aliyu, M. B., Iqbal, R. and James A. (2018). The Canonical Model of Structure for Data Extraction in Systematic Reviews of Scientific Research Articles. 2018 Fifth International Conference on Social Networks Analysis, Management and Security (SNAMS). 15-18 October, Valencia, Spain 2018.
3. Muhammad, B.A., Iqbal, R. and James A. (2019). Automatic Identification of Core sections in scientific research publication. A paper presented at the 2nd Doctoral Capability and Development Conference, Doctoral college, Coventry University, UK. Wednesday 1st May 2019, at the Elm Bank, Doctoral School.
4. Muhammad, B.A., Iqbal, R. and James A. (2020). Automatic Text Summarisation: What Size of a Automatically Generated Summary is Enough? A paper presented at the 3rd Doctoral Capability and Development Conference. Doctoral college, Coventry University, UK. Wednesday 25th March 2020 at the Elm Bank.
5. Muhammad, B. A., Iqbal, R., James A. and Nkantah D. (2019). Convolutional Neural Network for Core Sections Identification in Scientific Research Publications. 20th International Conference on Intelligent Data Engineering and Automated Learning. 16-19 November 2019, Manchester, UK.
6. Muhammad, B. A., Iqbal, R., James A. and Nkantah D. (2020). Comparative Performance of Machine Learning and Deep Learning in Text Classification. International Conference on Computing and Information Technology. 2020 International Conference on Computing and Information Technology, University of Tabuk, Kingdom of Saudi Arabia September 9-11 2020.

1.10 Chapter Summary

In this chapter, the background of the research has been outlined, including the research problem. The chapter also specifies the research questions to be addressed, the research motivations and purpose. The aim and objectives of the research have also been spelt out. A brief description of the methodology for the research as well as the original research contribution has also been stated. The thesis structure has also been outlined. Finally, a list of publications and presentations made in connection with this research has been included.

2 Literature Review

In chapter one, the research problem and aim, as well as the objectives have been clearly stated. In particular, the need for a framework or a unified approach to automate the data extraction stage in systematic review in the software engineering domain. In this chapter, a review of the related literature is reported. An automated search strategy, followed by a snowballing technique, was used to locate and retrieve the related papers on the subject. The search strategy was validated using already known relevant papers. Using this method, a detailed state-of-the-art literature on the related tools, strategies, technologies and methods relevant to the SLR automation was identified. The objective is to identify all the available tools that automate all or part of the systematic review process, including their underlying strategies and the degree of automation they provide. This chapter also reviewed the technologies and concepts that would enable the process to be achieved. The literature review has been an iterative process, repeated after every six (6) months. This is done to ensure up-to-date literature on the subject by capturing the missed or new research papers after the previous review.

2.1 Mapping Study

Reviews (of literature) of all kinds are meant to get as much information as possible to build a knowledge base for further research or inform decision making. The traditional literature review has some reliability issues, not only because they are necessarily inaccurate, but the selected studies were solely chosen by the writer. The researcher primarily decides what study to include or exclude using a simple rule: Only studies that support the rationale of the study are included (Briner and Denyer, 2012). Hence, they may be biased. Systematic review eliminates this bias through the rigorous process adopted. Both reviews deal with the substance of the research findings (Cooper, 2016). Mapping studies, however, do not involve the statistical analysis of the findings (Cooper, 2016). They are intended to explore how the given research topic is covered in the literature. In other words, they are intended to 'map out' the research topic rather than answer a detailed research question (Budgen *et al.,* 2008). Some factors, such as study identification, are common to both. However, they are different in their goals and data analysis approach (Petersen *et al.,* 2015).

2.1.1 Mapping Study vs Systematic Review

There exist commonalities between the mapping study and the systematic review. However, the mapping study may be more generic in scope, unlike the systematic review which aims to answer a given research question. Since mapping study is aimed at finding out how the topics are covered in the literature, having a good representative sample of the papers is more important than having many of them. Also, the quality assessment of the included studies is necessary in systematic review to determine the rigour and relevance of them. In mapping study, however, quality assessment is optional but may be important to ensure enough and efficient data (Petersen *et al.,* 2015; Wohlin *et al.,* 2013); hence the data extraction and synthesis in mapping study focus on classification and categorisation of the studies.

In this project, we performed the mapping study (review) of the literature with all the necessary rigour and quality assurance for a representative paper as possible.

2.2 Related works

As highlighted in chapter one, the SLR activity was first developed and used in biomedicine or healthcare to support evidence-based decision making (Guyatt *et al.,* 1992). With the rapid expansion of the evidence to be synthesised, the SLR becomes more complicated. This would

require new skills, including efficient literature searches and the formal methods of evidence synthesis.

The Cochrane collaboration provides current or emerging evidence available to guide decision making in the healthcare sector. The Cochrane database for systematic reviews (CDSR) is the leading source of the systematic reviews on the effect of healthcare interventions including the protocols for the reviews (Allen and Richmond, 2011).

Due to the robustness and the success of the SLR in healthcare, it was adopted in other fields of endeavour, including social sciences, crime and justice, software engineering etc.

The Campbell Collaboration produces SLRs on crime and justice, education and social welfare (Boruch *et al.* 2001). The goal of the collaboration is to develop, disseminate and update systematic reviews of studies on the effectiveness of social and behavioural interventions which are useful to policy makers, practitioners and the general public.

The Evidence for Policy and Practice Information and Co-ordinating Centre (EPPI-Centre) conducts and publishes systematic reviews on education, health promotion, employment, social care, and crime and justice. Committed to shaping policy and professional practice, with sound evidence, the EPPI-Centre develops tools and methods around systematic reviews and research synthesis, conducting reviews, supporting others to undertake reviews and provides guidance and training in this area. The EPPI-Centre is also involved in studying the research use. This includes the synthesis of research evidence to support decision-making in personal, practice and political endeavours. This provides the support to those searching for evidence to solve problems and providing guidance and training in this area (EPPI-Centre 2019).

In 2004, Kitchenham introduced *evidence-based software engineering* (EBSE) that pioneered the adoption of systematic review in software engineering research. EBSE was structured to integrate the research with industry in order to improve decision making (Kitchenham, 2004).

This research project is about the SLR automation (computer support) in the software engineering domain; hence the EBSE will be explored in greater depth. We explored some of the research works on automation or semi-automation or any computer support in any or all the stages of the systematic review process. This includes the tools, approaches and other techniques that support the process.

In all the domains, the basic idea of the SLR process is the same, and involves the retrieval, appraisal and synthesis of evidence (Van Altena, Spijker and Olabarriege, 2019). The

difference, however, lies in the nature of the evidence and how it is published. More differences are highlighted in section 1.2 in chapter one.

Being a structured process, a systematic review can take a full-time researcher from six (6) months to a year or more, depending on the expertise and experience (Van Alter, Spijker and Olabarriege, 2019). Also, with the ever-growing body of literature being produced daily and the increasing number of research questions that require answers, the system is surely overwhelmed. Automation has great potential to improve the process (Beller *et al.,* 2018). The work of Van Altena, Spijker and Olabarriege (2019) identified the need for automation in SLR. Automation may improve all the tasks in the various phases of the process from conducting the reviews to identifying the research gaps as well as writing and disseminating the review. That could cut down the duration taken to conduct review significantly, reducing it from years/months to weeks/days. Similarly, the work of O'Mara-Eves *et al.* (2015) concluded that bringing automation to the review process could reduce the reviewers time from between 30%-70%. For this reason, interest in automated systematic review is growing. Several tools and techniques have been advanced in this regard. Tsafnat *et al.* (2013) identified some potential technologies that would improve systematic review via automation. They identified the machine learning-based tools for screening the titles and abstracts in SLR evidence (papers). While the technologies seemed effective, however, they identified a number of areas that require more automation particularly in search strategy, abstracts screening, obtaining and screening the full text of the evidence and data extraction and synthesis. There are four (4) main tasks in the review process that require automation: retrieving the relevant evidence; evaluating the evidence; synthesising the evidence; and publishing the review (Beller *et al.,* 2018). These tasks have sub-tasks.

In general, the systematic review toolbox, a repository for tools to automate several stages in the systematic review process, has been identified by Marshall and Brereton (2015). This research carefully reviewed all the relevant tools and the approaches behind the tools in this 'box' and the report is detailed in the sub-sections below.

2.2.1 Support for Review Activity

Several tools for the systematic review process have been identified. In general, these tools can be categorised into two types (2): *general purpose* tools and *special purpose* tools. The basic or general-purpose tools such as word processors, spreadsheet packages, reference managers etc. were adopted to aid the process, and proved very useful (Marshall, 2016). Reference

managers include JabRef (JabRef 2019), RefWorks (RefWorks 2019), Mendeley (Mendeley 2019). Word processors include MS word, Google docs, WordPerfect, TextEdit etc. Spreadsheet packages include MS Excel, Google Sheets, and OpenOffice etc. These tools are not primarily developed for SLR tasks. However, they are adapted to provide some support in the process. The specific purpose tools, however, are specifically developed to support ((semi) automate) some or all the stages of the systematic review process. Tools to manage bibliographies also exist. Some of these act as meta-searchers, which allow searches to be done in digital libraries such as ACM or IEEE, or reference managers such as CiteSeer. They also permit the searches to be refined with their searchers.

In Biomedicine where SLR began, many tools and platforms support the review process. These include Abstrackr (Wallace *et al.*, 2012), EPPI-Reviewer (EPPI-Reviewer 2010) and RevMan (RevMan 2014). In addition, *Covalence* tool, developed by Cochrane collaboration, also supports the review process in biomedicine (Tsafnat *et al*., 2013).

2.2.1.1 Abstrackr

This is a semi-automated software for a predictive title and abstract screening of the studies included in the review process. It enables the abstracts of the included studies to be uploaded, screened and stored by the invited reviewers (Rathbone, Hoffmann and Glasziou, 2015). It is also an open source and online. OpenMeta is also an open source and cross-platform tool for performing meta-analysis in systematic reviews (Sadeghi and Treglia, 2017).

2.2.1.2 Covidence

A Covidence is a web-based tool designed to support the screening, data extraction and meta-analysis of the systematic review process (Babineau, 2014). It has a number of support features including importing citations, titles and abstract screening, uploading references, full-text screening, creating forms, risk of bias, data extraction, exporting the reports. The title and abstract screening is done using a keyword search. The desired keywords are highlighted and voted by the reviewers. The reviewers do the full-text screening by reading the entire text and then deciding which one to include or not. The reasons for such decisions are also captured and stored. Data extraction 100% manual. Reviewers have to fetch the desired data from the text. The data elements are categorised into population, intervention, comparison and outcome (PICO) elements. They also serve as a guide for what data to extract. Covidence provides the forms to store such data. The Covidence also supports the risk of bias assessment. The problem with this tool is that it requires a license to be used. However, it is free to use only for Cochrane authors. Trial versions are available also but for a limited time, less than a month.

2.2.1.3 EPPI-Reviewer

This is a multi-user, web-based application for managing and analysing data for use in research synthesis. It also enables collaborative research by bringing researchers in separate locations together. The key functionality EPPI-Reviewer includes: Reference management: EPPI-Centre manages thousands of references obtained from comprehensive searches of electronic databases which can be imported in a variety of formats by checking out duplicates either manually or automatically using 'fuzzy logic'. It also supports the storage of the documents from these references in different formats such as .pdf, doc etc. It also supports Study classification and data extraction. It supports the classification of the studies by defining a flexible coding scheme for inclusion/exclusion/eligibility criteria, descriptive mapping of research studies and capturing the details of the study. They also support plotting results, generating reports and applying certain meta-analysis techniques. Study synthesis is to some extent supported by the software. Using the text mining functionality, it provides automatic document clustering by identifying the significant terms in the documents that have already been included. It also allows line by line coding of the textual data. This is done via the organisation and structuring of the codes graphically into 'conceptual relationship diagrams which displays the analytic and descriptive themes in the code. The EPPI-Reviewer also supports review management. The software provides support for unlimited sharable reviews. It also allows the allocation of the tasks to individual reviewers/users. This includes classification tasks such as assigning users to responsibilities such as data extraction or studies screening. Users can also report/track their progress automatically through the review charts that update automatically.

2.2.1.4 Review manager (RevMan)

This is a software for the production and maintenance of Cochrane reviews (The Nordic Cochrane Centre 2014). It facilitates the preparation of protocols and full reviews, including text, characteristics of studies, comparison tables, and study data. It can perform a meta-analysis of the data entered and presents the results graphically. RevMan is also efficient for the diagnostic test accuracy studies, reviews of studies of methodology and overviews of reviews. The latest version is Revman 5. Web-based RevMan is also available, featuring an improved and modernised workspace and user interface, RevMan Web offers a more intuitive and enjoyable user experience than RevMan 5. Although this software is no longer being developed, the support for the reviewers who use it is still available from the developers.

2.2.1.5 RobotReviewer

RobotReviewer is an open sourced, machine learning system and NLP system that semi-automates biomedical evidence synthesis. It works on full text journal articles describing the RCTs (Marshal, Kuiper and Wallace, 2015). It also appraises the reliability of RCTs as well as extracts the text describing the key trial characteristics. Using this information, the RobotReviewer then generates a report automatically. Using the Cochrane Risk of Bias (ROB) tool, the RobotReviewer determines the risk of bias in the clinical report. The items in the ROB tool include random sequence generation, allocation concealment, blinding of participants and personnel, blinding of the outcome of the assessment, incomplete outcome data and selective outcome reporting.

2.2.2 Methods/Tools Support for Systematic Review in Software Engineering

The various methods that provide automation in software engineering as well as their implementations (built as tools) have been identified. This includes both general purpose and special purpose tools. We reviewed them, including the prescribed approach behind the tools. The special purpose tools include: SLR-TOOL: A Tool for performing Systematic Literature Reviews (Fernández-Sáez,, Bocco and Romero, 2010); SLuRp – A Tool to help large or complex systematic literature reviews deliver valid and rigorous results (Bowes, Hall and Beecham, 2012); StArt (state of the art through systematic review) (Hernendes, 2012); SLRTOOL: A tool to support collaborative systematic literature reviews (Barn *et al*., 2014); SESRA - A web-based automated tool to support the systematic literature review process (Molleri and Benitti, 2015); and Parsifal Tool (Parsifal, 2015). These tools provide support for the overall review process. Other specific purpose tools, however, support for only specific stage(s) of the review process. These include: REviewER: (REviewER 2017), ResearchR (SR ToolBox 2016), Rayyan (Ouzzani *et al*., 2016) etc.

Marshall, Brereton and Kitchenham (2014) reported that the best all-stage specific purpose tool is *SLuRP* and only accounts for 43% automation support for *conducting* stage and 61.8% support for other auxiliary processes. This shows the need for further improvement of the tools, especially for the conducting stage. Moreover, our assessment of the tools not just confirmed the Marshall, Brereton and Kitchenham's research but also revealed that most support provided by all the tools, without exception, at all stages is semi-automated. Reviewers still have to put in a huge effort and time to realise the process.

In the subsequent subheadings, each of the all-stage general-purpose tools are explored to identify the areas of strength and weakness and, hence, where the need for more research is critical. For each tool, a very brief description is given, followed by the functional supports (automation) it offers. Finally, the weakness of the tool that requires further automation is highlighted.

2.2.2.1 The Software Engineering Systematic Review Automation (SESRA)

This is a web-based tool developed by Molleri and Benitti (2015) to support all the stages of the systematic review process in software engineering research. The tool implementation follows the guidelines proposed by Kitchenham and Charters (2007). Specifically, SESRA allows the researchers to conduct the SLR activity through the three (3) distinct phases: planning, conducting, and reporting. The various support SESRA offers to the review process would be reviewed in the light of the phases of conducting the SLR process which is detailed in section 1.1.2 above.

Planning phase: The planning phase consists of various stages and activities. These activities are supported by SESRA as follows:

Commissioning of the review: This stage is well supported by the SESRA. It allows the review team members to be defined with an email notification sent to each included member. This follows the development of the review protocol, which includes other substages.
Protocol Development: Research questions formulation: The research question is normally broken down into population, intervention, comparison and outcome (PICO) components. This is essential for taking a focused clinical question. In software engineering, however, the PICO elements are adopted to reflect similar, but not exact concepts. For example, the population may stand for a process or problem, intervention for a possible solution, comparison for a current practice or opposing viewpoint and outcome for measuring what works best. The components are then assembled to generate the search string.
The search string also involves the generation of the synonyms of the various components of the PICO elements. SESRA supports the PICO generation although it does not automate the PICO elements completion from the research questions. Reviewers manually enter the PICO elements. From the PICO elements entered, the search string is automatically generated. The search string, however, is not specific to any bibliographic database engines.

SESRA supports specifying the inclusion/exclusion criteria. Although it does not have any pre-defined set of inclusion/exclusion criteria, it nonetheless allows the reviewers to set their criteria. Each criterion is categorised under a separate heading. The criteria are used to select studies into the review process. The tool does not apply the criteria automatically, thus allowing them to be applied by the reviewer. The information sources are partially supported by the tool. It presents a list of possible sources such as ACM Digital Library, IEEExplore, Google Scholar, ScienceDirect, Springer, Institution of Engineering and Technology etc. In the preliminary experimentation by this research, however, the only libraries found to be supported are Springer, IEEExplore and Institution of Engineering and Technology.

SESRA also allows the reviewers to specify any other sources of the papers. The data about the primary studies is not extracted automatically. It is manually extracted and put into the forms defined in the protocol. Therefore, the data extraction forms are defined. The tool has partial support for this.

Conducting the SLR begins with the identification of the research papers. SESRA has very little support for this phase. Although it does attempt to search for papers automatically but only downloads the link to the paper. The researchers (not SESRA tool) undertake the selection process based on the title and abstract, and a copy of each paper must be obtained to assess the full text.

The tool allows the reviewer to import (details of) a new study either through a form where the user manually enter the details of the study including the abstract or importing bibText file which contains all the details of the study.

In summary, the support/automation for study identification in SESRA is very limited. It does not download the paper nor are the details of the study downloaded (except the link to the paper). The assessment of the quality of the included studies was based on some selected method (guidelines). The method or approach used in SESRA include: The CRD's DARE criteria, Kitchenham's quality assessment checklist (Kitchenham *et al.*, 2007) and Quality Assessment Checklist (Dybå and Dingsøyr, 2008). SESRA also allows the reviewers to define their guidelines for the quality assessment. These guidelines assess the quality of the studies using two scales: the Boolean (Yes/No) and an array of values e.g. 1-1-5 scale. The study with a higher score should be more relevant to the research question.

The reporting stage of the systematic review is mostly a manual process. The only support it renders is the provision of the forms into which the report is formatted.

The SESRA tool supports nearly all the stages of the systematic review activity but it partially implements the automation for the conducting stage, leaving reviewers to perform lots of manual bits of the activities in the process, especially the studies selection, quality assessment and data extraction. In conclusion, although SESRA does provide the support for managing the process, many manual requirements remain.

2.2.2.2 StArt (State of the art through systematic review).

As discussed above, the SESRA tool has major weakness in the conducting stage, which is the most labour intensive and repetitive process (Hernendes *et al.,* 20102). The **StArt** tool, developed by LaPES (a laboratory for search in software engineering), was designed to provide more support/automation, particularly in the conducting phase. StArt follows similar guidelines (approach) used by SESRA. It begins with the protocol definition allowing the reviewers to specify the objective of the review, the sources to be searched, the quality form, inclusion/exclusion criteria, keywords and synonyms etc.

The identification of the primary studies is not supported by the tool. The reviewers externally identify the relevant studies and then import them into the tool using formats such as *bibtex, medline, RIS and Cochrane* reference files. The studies can also be imported manually using a designed form. After importing the reference files, the details (such as paper id, Title, Author, year, score etc.) are used for onward processing.

The tool provides a score for each study using the keywords specified in the protocol. It uses the keywords to score the studies according to the number of occurrences of these words in their title, abstract and keywords. This score suggests the studies relevance order (Hernendes *et al., 2012*). The score is automatically calculated reflecting the number of times the keywords, defined in the protocol, occur. Other attributes of the studies imported into the tool have to be manually populated by the reviewers. For example, the status and the reading priority have to be set by the researchers after reading the full text of the article.

The selection of the relevant studies for inclusion into the review is done manually and using the score citation automatic (SCAS) strategy. This strategy is based on two (2) features: First is the score which supports the analysis of primary studies by the frequency of occurrence of search string terms in the title, abstract, and keywords. Second is the number of citations within the same review. The SCAS strategy groups the studies into four (4) quadrants: Q1, Q2, Q3, and Q4. Q1 is a group for studies with a high score and at least one citation. Q2 have a high

score but no citation. Q3 have a low score but at least one citation. Q4 have low score but no citation at all. Q4 are automatically rejected, Q1 are automatically accepted. Q2 and Q3 have to be reviewed manually. After this stage, the final list of papers is selected.
Having paid more attention to the execution phase of the systematic review, StArt tool leaves the search for relevant studies to the users. Any primary studies found by the user are the ones used for the review exercise. The relevance score is calculated based on the studies available (obtained by the reviewers). The score is based on the title and abstract, but this may not reveal the relevance of any studies in a systematic review. This is because abstracts in software engineering are often poorly crafted (Budgen *et al.*, 2007). Therefore, the need to dig deep into the content of the studies is paramount.

After the selection of the primary studies based on their relevance depicted by the score, the support for the review, by this tool (StArt), nearly comes to an end. But the primary goal of the SLR process is to answer the review question by analysing the full text. Another goal is to identify the research gap. StArt tool does not support research synthesis (full-text extraction and analysis).

2.2.2.3 SLR-TOOL: A Tool for Performing Systematic Literature Reviews

This tool, a multi-lingual (provides both English and Spanish interfaces), is also a web application designed using Java programming language. It also adopted the SLR guidelines proposed by Kitchenham and Charters (2017). The tool is also freely available for use by any reviewer or researcher.

As a result of the complex search string limitation of the major search engines, the tool does not generate search strings since it does not perform the searches, but enables searches to be made and refined using the Lucene search engine (Fernandez-Saez, Bocco and Romero, 2010). In other words, this tool does not support searching for studies but allows the refinement of the searches.
An important feature of the tool is the screening of the primary studies through categorisation and sub-categorisation which are aimed to easing the synthesis and analysis of the data. The tool uses a two-level classification scheme for the categorisation and sub-categorisation of the studies.
SLR-TOOL also uses text mining (clustering technique) to cluster documents. This is done using the similarities within the documents. The aim of the clustering is to determine if the

documents retrieved during the search from the search engines are relevant to the subject/questions/query. The irrelevant documents are excluded at this stage.
The categorisation is used to generate tables and charts used to summarise the data. This provides the researchers with the visualisation of the review. SLR-Tool exports all the data collected in the review process to Excel file sheets. This makes the data more manageable, allowing them to be used in any documents or papers in which it is intended to report the conclusions obtained. SLR-Tool also allows all the bibliographic data from the primary studies uploaded in the tool to be exported to the format accepted by bibliographic packages such as EndNote, bibtex and Ris. This facilitates the use of these references in subsequent publications. A big weakness of the tool is its lack of support for meta-analysis to integrate empirical data from the primary studies.

2.2.2.4 Parcifal Tool

This is also an online tool for systematic review within the context of software engineering. It is designed for geographically distributed researchers to work together within a shared workspace. The strength of this tool is in the conducting stage of the process.

During the planning phase, Parcifal offers the interface to state the research questions which the review sets out to address. From the research questions, PICO elements could be specified. The PICO elements are specified by the reviewer. From the PICO entry, the search is automatically generated. Parcifal also allows the keywords and synonyms. Sources of the search are also specified either choosing from a pre-defined list (IEEExplore, ACM Digital library, ScienceDirect, Scopus, ISI web of science, EI compendex, Springer) or suggesting self-sought sources using the designed forms. The inclusion/exclusion criteria are also well supported, although it is left to the user to suggest the criteria which are important to the review. It also provides mechanisms to build a quality assessment checklist and data extraction forms. The quality assessment in Parcifal is not based on any guideline for conducting the systematic review exercise. Rather it is based on the questions the reviewers have generated. The tool provides answers to the questions using a scale of three (3) answers, depicting how well the study answers each question. The scores are assigned to each question. For example, ‘Yes’, ‘Partial’, ‘No’ with 1.0, 0.5 and 0.0 scores respectively (the scale is editable).
Conducting the systematic review exercise is the primary task which any tool should aspire to support/automate (although planning phase is important to the conducting phase). This stage

begins with the identification of the research. During the conducting phase, the Parcifal tool enables the automatic search of publications using the search terms developed in the protocol. However, selecting the studies only concentrates on the *bibtex* files imported. Duplicates can be removed automatically but the tool relies on the users' judgement as it does not automate this task but provides the button to classify if the study is duplicated or not, from all the different sources searched. Executing the quality assessment is supported through answering the questions developed in the protocol of the planning phase. Each study is evaluated using the scale provided. As highlighted above, the studies are presented in respect of their score in descending order, which is visible at the data extraction phase (the support for this is limited in Parcifal). Note that the assessment fully relies on the full text of the paper which the user must have read, from which he/she answers the question.

Data is extracted from the papers by the reviewers. The only support the tool offers, is the provision of the data extraction forms which must have been developed in the planning phase. The automation provided by Parcifal is very limited.

2.3 Screening Strategies/Approaches

Each piece of evidence (primary study) identified for the review must be appropriately appraised. Weighing up each evidence ensures that small or irrelevant findings do not get interpreted or weighed wrongly. This process must be rigorously executed. For example, the results from small or badly designed studies should not carry as much weight as the robust and substantial studies. This process summarises the most relevant literature to identify commonalities and differences between them. This way, the overall picture of the review can be established in response to the review question and identify research gaps where new knowledge and research would be helpful. To achieve this goal, therefore, an efficient strategy must be used. So far, several strategies to establish the quality of the individual primary studies have been developed. A few tools, identified and analysed above, adopted some of these approaches to screen the relevant literatures. The following subsections describe some of the approaches.

2.3.1 Systematic Literature Review based on the Visual Text Mining (SLR-VTM).

This approach, developed by Tomassetti *et al.* (2011), supports study selection/screening during the conduct of SLR process. It was implemented using a visual text mining, an extension of the text mining technology, and the information visualisation techniques that supports visualisation and exploration of data (Felizardo, 2012). It is comprised of three visual

representations of the primary studies. These are: (i) a document map; (ii) edge bundles and (iii) a citation network. The document map creates a 2D representation of the primary studies enabling users to view the similarity relationship between the studies. A document map is created through the conversion of all primary studies into multi-dimensional vectors based on all the terms extracted from the studies. Cluster technique is used to screen studies by identifying the regions from the 2D with similar content in terms of their title, abstract and keyword. Edge bundle is used to visualise the number of times a document is cited. Papers that have many citations are regarded better candidates. References which the papers use are also considered by this strategy. This is achieved by the generation of the citation map, which puts together the common references shared by all the studies into a map. The citation network is constructed showing the circles (primary studies) with their cited references, the primary studies' references and the connections between papers via the set of references that they share. Through this depiction it is possible to see citations between the primary studies with their references and references of other primary studies. Reference lists from relevant primary studies could be another source of evidence to be searched. Hence, papers that share references with a relevant paper could be more appropriate for inclusion in the SLR. On the other hand, primary studies that are not connected to any other studies (i.e. do not share citations or references), are more likely to be irrelevant documents in terms of the research question and may be more readily excluded from the review.

Visual text mining was also reported by Malheiros *et al.* (2007) to build documents similarity in the systematic review using the projection explorer (PEx) tool. This technique, however, constructs the citation network. PEx provides two (2) methods to identify and classify the studies. One method pre-processes the raw ASCII version of the article to build a vector space model (VSM) of documents from selected terms. The similarity between two documents is computed using the cosine distance between their vectors. The other method follows from the raw ascii files using the normalised compression distance (NCD). The result from the two (2) methods was projected using 2D maps.

The use of the visual data mining techniques in the conduct of the systematic review offers a compact way of viewing information to show and compare the relationship between documents. However, this technique comes with its own cost and challenges. First, the visualisation technique uses graphics such as icons, images, symbols etc. that require lots of resources and time to make. In other words, graphics are time and resource intensive. It also requires an interpretation of the various symbols used. This creates another overhead of

creating the interpretation. Users who do not have the skill to interpret the graphics are left in deadlock. Also, even for those with graphics interpretation skills, more time and resources are needed to produce and use these interpretations. One of the very aims of (semi) automating the task of SLR is to reduce the time required to undertake it.

Poor interpretation of the graphics is another pitfall that poses a greater challenge in adopting this technique. Misinterpretation of the data from the symbols would render the whole exercise vague. Also, poorly generated graphics are very hard to interpret (DM&E 2016). This is because, graphics rely on symbols and different people interpret symbols differently depending on their background. This could be the very reason that most of the tools supporting Systematic reviews do not support this approach.

The implementation of this technique combines text mining algorithms with interactive visualisations to help the user make sense of a collection of documents without actually reading them. This means that the quality assessment of the primary studies was superficial, because only metadata details of the studies such as title, keywords etc. are considered in the evaluation of the studies. The relevance of each study was based on the frequency of keywords and the intersection of the common keywords within the entire collection of studies.

The implementation of this technique is also partially semi-automated. The tool worked by using an already prepared spreadsheet. In this sheet, the details of the primary studies were extracted by the users and inserted into the xlxs sheet. This follows the conversion of the pdf documents.

2.3.2 Search String Simulation Strategy

Two strategies were developed by Abilio *et al.* (2015) to screen the primary studies by considering the terms used in the search strings. The first strategy is based on information retrieval (vector space model) which ranks the primary studies in decreasing order of importance based on the weight of the evidence. The second strategy executes the relevance classification according to the terms used in the search string. The frequency of the search terms used by the reviewers is calculated from the title, abstract and the keywords contents. This is done in the form of simulation of the Boolean expression used in the search string. The two strategies were implemented through an algorithm to support the SLR process.

While the strings are effective in the searching and identification for the related studies, the search string frequency does not confirm the relevance of the study under investigation. The

frequency of the search terms in the title and abstract are, of course, a key factor for the study identification but it is of little value to determine their relevance. Relevant studies must be cited as frequently as possible by other studies. This approach does not have much support as no tool allows the reviewers to use or evaluate the proposed strategy.

2.3.3 SCAS strategy

The score citation automatic strategy (SCAS) was put forward by Octaviano, Silva and Fabbri (2016). This strategy determines the relevance of primary studies based on two factors: score and citation. The score is calculated as the frequency of the search string terms appearing in the title, abstract and keywords. The studies with score are potentially considered relevant while studies with low score are considered non-relevant and hence, rejected. Two techniques: 50% and J48 decision tree, are used to determine the cut-off values for a high or low score. The studies are organised in descending order by score. The score of the study in the middle of the list serves as the cut-off value score (50%). The J48 decision tree technique was implemented using the WEKA tool.

The citation is determined by the number of times a particular study is cited by other studies within the same SLR project. These two complementary features combine to semi-automate the study selection phase using text mining techniques. SCAS strategy was implemented by the StArt tool identified and discussed previously. However, the StArt tool cannot access the content of the studies directly, but rather relies on the bibliographic data of the studies which are imported through the *bibtext* file (other file formats are also allowed such as RIS etc.). The *bibtex* file containing the bibliographic information of the studies searched from different search engines is imported into the tool. The *bibtex* file containing the bibliographic information about the studies searched from different search engines is imported into the tool. The *bibtex* file contains information as: title of the study, year of publication, author, publisher, journal, keywords, and ISBN etc. The bibliographic file does not, however, contain the abstract text of the study. To get the text of the abstract into the tool, users have to manually access the study and extract its contents for onward manipulation.

The problem with this approach is that both the score and citation could not be used to fully determine the relevance of the studies, hence, the selection. First, studies with high citation count are those that have been published for some time. This means that studies that are recently

published tend to have low/no citation count because it takes time to search, use and cite studies by several potential researchers. By this approach, therefore, recently published studies could be rejected no matter how relevant they could be. This factor is of great consequence on the overall activity as relevant studies could end up being thrown away simply because of the low citation. Also, this approach is subjective. The score, which is derived from the search string does not, in any way, predict the findings/result of the study. It only shows the area of interest which the study investigated, but not what the study finds or concludes. Therefore, a deeper look into the relevant contents of the study is needed.

Also, the studies with shorter titles are cited more often than those with longer titles. Furthermore, the studies with results-describing titles are cited more often than those with method-describing titles (Paiva, Lima and Paiva, 2012). Again, since most of these tools do not have access to the studies, they rely on the *bibtex* file containing the bibliographic information about the studies. Greenhalgh and Peacock (2005) reported that they only identified 30% of the literature through electronic sources. With 30% recall, users have to manually search from other non-electronic sources to identify more relevant studies. The bibliographic information on these manually identified studies has to be entered into the *bibtex* file.

2.4 Data Extraction

To answer the research question, the data (research findings) must be extracted from the primary studies. The data concerns the methodology used, study type, results obtained, conclusions reached etc. (Fernandez-Saez, Bocco and Romero, 2010). SLR deals with the substance of the research evidence, focusing on the results of the primary studies and discussing the findings (Cooper, 2016). The data extraction is the most challenging stage in the SLR activity. It involves the extraction of the summary of findings (main results), main topics covered, main methods used or proposed etc. (Kitchenham and Brereton, 2013). From our experimentation and assessment, the tools have very little support for automatic data extraction. The table 2.1 below shows the support for the data extraction for the various tools. The criteria, including the desired features, for evaluating the automation strength of the tools was developed by Marshall, Brereton and Kitchenham (2014). The scale was used to score each approach/tool with respect to the support it provides:

The activity is fully supported - 1

Support is limited or some aspects of the activity are not supported - 0.5

No support at all - 0

Table 2.1 Functionality summary of the methods behind the tools

Method (Approach)	SESRA	SLR-TOOL	SLRTOOL	StArt	SLuRp	Parsifal
Gen. of Search string	**1**	**0**	**0**	**0**	**0**	**1**
Automated searches	**0.5**	**0.5**	**0**	**0.5**	**0.5**	**0.5**
Local Importing studies	**0.5**	**0.5**	**0**	**0.5**	**0**	**0.5**
Study Selection/validati	**0**	**0**	**0**	**0.5**	**0**	**0.5**
Quality assessment	**0.5**	**0**	**0**	**0.5**	**0**	**0.5**
Data extraction	**0.5**	**0.5**	**0**	**0**	**0**	**0**
Text analysis	**0**	**0**	**0**	**0**	**0**	**0**
Report Writing	**0**	**0**	**0**	**0**	**0**	**0**

As shown in table 2.1 above, there is very little support for automatic data extraction from the tools. The only support available is the provision of the data extraction forms defined in the protocol. Using the data extraction forms, reviewers manually extract the data to populate the forms. Hence, the need for more research efforts in data extraction automation. The details of the experimentation can be found in the appendix E.

2.4.1 Data Extraction: Progress Made

To date, there has been a significant research effort in this direction. Some research works performed data elements categorisation. These include the work of Kim *et al.* (2011) who performed sentence classification into one of PICO categories. They used Conditional Random Fields, a machine learning algorithm. Lexical, syntactic, structural and sequential information were used as features for the classification. Similar work was done by Boudin et al. (2010), but using a different method. They used a combination of Random Forest (RF), Naïve Baiyes (NB), Support Vector Machines (SVM) and Multi-Layer Perceptron (MLP) supervised machine learning methods to perform the task using MeSH semantic types, word overlap and number of punctuations as features. Hara and Matsumoto (2007), Chung (2009), Huang *et al.* (2011),

Verbeke *et al.* (2012), Robinson (2012) Huang *et al.* (2013) and Hassanzadeh *et al.* (2014) performed a similar task using similar techniques. However, Verbeke *et al*. (2012) used a statistical learning-based approach (*Klog*), a different technique from the previous works. Various accuracies and F-scores were reported by these studies. In general, all these studies reported a good result on the identification of data elements in abstract only but did not extract the data. However, all these works performed the classification at the sentence level using only the abstracts.

Another research study worked on the full text documents of the evidence (primary studies). However, just like in the previous category, the data elements were identified but not extracted. These studies view data extraction as a classification task. This has effectively been done in a number of studies. Zhao *et al*. (2012) performed a two-way classification scheme: one at the sentence level and one at the keyword level to obtain patients' details. They used CRF and about 20,000 medical abstracts and full text articles from 17 journal websites. Similar work by Hsu *et al.* (2012) identified the presence of words such as "hypothesis", "statistical method", "outcomes", or "generalisability" in a sentence. Liakata *et al.* (2012) used machine learning to identify the Core Scientific Concepts (CoreSec) which include: Hypothesis, Motivation, Goal, Object, Background, Method, Experiment, Model, Observation, Result and Conclusion. They reported a high degree of accuracy (up to 76%). However, the experiment was performed at the sentence level. Sentences were manually extracted and used to train the machine learning models to recognise the core scientific concepts. Song *et al.* (2013) used Maximum Entropy classifier(MaxEnt), SVM, MLP, NB and radial basis function network (RBFN) for sentence classification into one of four (4) groups: analysis (statistical facts found by clinical experiment), general (generally accepted scientific facts, process, and methodology), recommendation (recommendations about interventions), and rule (guidelines). Information gain (IG) and genetic algorithm (GA) were used for feature selection. Marshall, Kuiper and Wallace (2015) used soft-margin support vector machines for risk of bias assessment. These works only identified the data elements without extraction from the full text articles.

Notwithstanding, a number of research efforts did explore the data elements extraction from both abstract and the full text of the documents. Works that extract data elements from abstract only include the work of Kelly and Yang (2013) who used regular expressions to identify the number of participants, age and ethnicity included in the study characteristics. This was in line with the work of Madsen *et al.* (2008) who identified the number of participants using a SVM.

Similar work was done by Xu et al. (2007), Summerscales *et al.* (2009) and Summerscales *et al.* (2011).

Data extraction attempts from the full text document are also documented from some research works. The ExaCT tool was developed by Kiritchenko *et al. (*2010) to assist the users in locating some data elements such as sample size and eligibility criteria. Other works such as De Bruijn *et al.* (2008), Lin *et al.* (2010), Zhu *et al.* (2012) and Restificar and Ananiadou (2012) all worked on full text to extract data elements using different methods. All these worked on the RCTs to identify data patient's information. It was implemented as a text classification task using manually prepared sentences.

2.5 Research Gap

From the various works reviewed, the summary of which is shown in table 2.1 above, we can conclude that the automated extraction of relevant information from SRDs (documents) for use in text processing tasks such as the SLR process has not yet been achieved. Hence, this task remains a very challenging task in SLR process. Jonnalagadda *et al.* (2015) concluded that there is no unified framework for information extraction from the SRDs. Previous research works focused on the extraction of few/limited number of data elements. In other words, the lack of a unified framework is the main obstacle to the automation of the data extraction from SRDs. The absence of a viable structure or framework is the missing piece in the data extraction automation of the SLR process. Any good and efficient approach must take into consideration the underlying format of reporting in the research publication.

Also, Jaspers, De Troyer and Aerts (2018) reported that the automation of the data extraction task from SRPs (for use in SLR) is still not feasible. In particular, the natural language processing (NLP) techniques have not been explored to fully or even partially to automate data extraction in systematic reviews. Hence, the tool to automate the data extraction procedure is not feasible currently.

Therefore, in consideration of the above points, more research effort is needed for the most time-consuming, most challenging and the longest stage of the SLR activity which is the data extraction stage. Such research effort should also focus on a unified framework or approach for the execution of this task.

Similarly, the metrics for evaluating the efficiency of machine learning models are only machine focused. Human actors are not involved. Hence, the machine model's predictive power would significantly improve if human expertise is incorporated to the process.

2.6 The Technologies

The technologies relevant for the full automation of the review process have not been fully explored. Such technology has advanced over time enabling the discovery of the knowledge and synthesis, but this has not been applied to SLR. The idea of using such technology has been muted a long time (Rennels *et al.,* 1989) but it has not been fully developed and exploited. The relevant state-of–the-art technologies that can support automated SLR are described below.

2.6.1 Text and Data mining

Text mining is defined as the process of discovering knowledge and structure from unstructured data by automatically extracting information from written sources (Hearst, 2003). The goal of text mining is to turn text into data for analysis, identifying associations among entities and other information in text. This makes it significantly different from data mining which tries to find interesting patterns from large databases. In other words, text mining extracts information from natural language (unstructured) text rather than from structured databases.

Text mining technologies such as natural language processing (NLP) and machine learning (ML) have proven to be efficient in data processing including synthesis and making sense from a body of literature. However, Jonnalagadda *et al.* (2015) reported that NLP techniques have not been fully explored to fully or even partially automate the data extraction in systematic review. Some studies have extracted elements through NLP, but the elements were limited in number to seven (7). There remains the need for a unified framework for data extraction in SLR and there is clearly space for further progress.

The choice of text mining technologies reduces the time taken for the review by the facilitation of identification of relevant literature, rapid description or categorisation and summarisation. Text mining is a well-established practice commonly used to extract patterns and non-trivial knowledge from unstructured documents written in a natural language (Tan, 1999). Text mining technologies include automatic term recognition, document clustering, classification and summarisation to support the identification of relevant studies in systematic reviews (Ananiadou *et al.,* 2009).

Text mining algorithms have been applied to systematic review in some projects. O'Mara-Eves *et al.* (2015) conducted a systematic review on the use of text mining in identifying relevant studies for inclusion in a systematic review. They identified a number of text mining technologies used in the screening stage of the systematic review; they include SVM, active learning, NB and Complement NB algorithm, visual text mining, semantic models, EvoSVM, visual data mining, k-nearest neighbour and Latent Dirichlet Allocation. Big data analytics and various application areas have been discussed in Iqbal *et al.*, (2020) and Iqbal *et al.*, (2020). These technologies have been used for the performance of text mining processing in the systematic review activities. Some of the state-of-the-art machine learning algorithms relevant to the research are as follows.

2.6.2 Machine Learning

Machine learning is a branch of artificial intelligence (AI) that powers systems with the ability to learn and improve on a given task without explicitly being programmed to do so. The computer programmes are designed and trained to automatically learn the patterns from the data set without human intervention (Jordan and Mitchell, 2015). Existing research also shows the use of machine learning approaches for text classification (Pin, Yuming and Chang 2020), emotion detection (Gupta *et al.*, 2019; Karyotis *et al.*, 2017). Machine learning algorithms include supervised, unsupervised, semi-supervised and reinforcement algorithms. This research employed the supervised approaches because we have a fixed number of classes and training data. The following subsections discuss some of the machine learning algorithms used in this project. The choice and justification for the choice of the ML algorithms is fully discussed in section 3.5.

2.7 Deep Neural Networks

Deep learning has gained prominence for text classification tasks in recent times. The neural networks based deep learning models map the words in a text to vectors. These vectors are then mapped into a fixed length representation. Different neural networks models exist such as the convolutional neural network (CNN), recurrent neural network (RNN), Recursive neural networks (RecNN) etc. In this project, CNN is chosen because they are faster computationally than the recurrent neural network (RNN). One reason why RNN are computationally slow is that each word in the string has to be processed sequentially. In contrast to CNN, however, all

the elements in CNN are simultaneously processed. This speeds up processing immensely (Dauphin *et al.*, 2017).

The CNN has initially been used for image processing and, hence, been a major break-through for image classification. However, it has also been applied to NLP tasks and has proven to be effective, particularly in text classification. Kalchbrenner, Grefenstette and Blunsom (2014), Kim (2014) etc. have reported an excellent result for NLP tasks such as sentence classification and relation classification. Hence, the model was chosen for use in this research. Kim (2014) performed questions classification using CNN. In general, questions are made of short text using not more than a sentence. Similarly, Vu *et al.* (2016) performed relation classification in CNN. Also, Kalchbrenner, Grefenstette and Blunsom (2014) developed a *k*-max pooling for sentence modelling.

2.8 Summarisation

The volume/availability of data is exploding at an exponential rate. The data is ever-growing. Data does not necessarily mean information, let alone knowledge (McCargar, 2004). This spikes the demand for automatic summarisation. About 2.5 exabytes of data is produced everyday (Marr, 2018). In biomedicine alone, more than 1 million papers are dropped into the PubMed database every year. That is about 2 papers per minute (Landhuis, 2016). It also does not have a pre-defined data model, nor is it any organised pre-defined model. Reducing the volume of the text is essential to extract the needed data and information. To harness the information within this ever-growing volume of text, robust methods and approaches must be explored. Automatic summarisation reduces such volume of information to a few lines of information which is more easily 'consumable' by humans or machines. The summary of a document is a reduced but precise representation of the text which seeks to render the exact idea of its contents. Its principal objective is to give information and provide privileged access to the documents (Toress-Moreno, 2014). Human or computer can generate summaries. Summarisation is automatic when generated by the software or algorithms. It generally involves selection (compression) of the text and discarding the rest of the text. The discarded part of the text is not considered relevant to the summary. How do we select the relevant text from the volume of text? This is one of the major questions in automatic text summarisation, i.e. finding the relevant information to be included in the summary. The automatic summarisation algorithms can be more effective than human summarisation. They are faster

(easier), less biased and can improve the effectiveness of indexing. Furthermore, automatically generated summaries do not need to be stored online as they can be generated online as needed, in accordance with user requirements (Moens, Uyttendaele and Dumortier, 2000).

2.8.1 Types of Summarisation

Summarisation is one of the applications of NLP. It is also undoubtedly an interesting but challenging process in the field of NLP. It involves generating the main (meaningful) points of the text in a precise and concise form from a single or multiple text sources such as books, news articles, blog posts, research papers, emails, tweets, Facebook posts etc. The way human intelligence summarises a text may entirely be different from the way machines (artificial intelligence) attempt to do the same. Humans read the entire texts to develop understanding of it and then try to write the summary highlighting the main points in the original text. In doing so, the human can completely use a different choice of words/sentences/expressions than those used in the original text (Allahyari *et al.,* 2017). Humans can also reuse part of the sentences. Accomplishing such task with machines is undoubtedly a daunting task as the machines lack such intelligence similar to human's (knowledge and language capability). Hence, automatic summarisation is not a trivial task. There are two types of summarisation: abstraction and extraction. Abstraction is the way humans do it and is difficult for machines. Extraction is the more usual approach for machines and involves selecting the most relevant sentence from a text.

2.8.1.1 Extractive summarisation

Extractive summarisation involves pulling out the most important bits from the original text and then putting them together to form the summary. This involves re-using some of the sentences or phrases (Khatri, Singh and Parikh, 2018). Occurrences of words or sentences or phrases are counted and analysed based on their frequency and location of their occurrence in the source or original text. In the extraction, therefore, the exact key sentences and phrases are pulled out with little or no modifications from the original piece of text and stacked together to form a summary. In other words, the approach chooses a subset (most important) of the sentences from the original text. The new (extracted) sentences represent the most important information from the original document (Barzilay and Elhadad, 1999). In the extractive method of summarisation, therefore, identifying the right sentences is of utmost significance. This approach is sometimes dubbed knowledge poor approach and relies heavily on the term

weighting algorithms and methods of information retrieval (IR) (McCargar, 2005). This is also known as the statistical text extraction approach.

Extractive text summarisation, therefore, involves three (3) stages: 1. Intermediate representation of the sentences to depict the topics covered by the sentences; 2. Scoring the sentences based on the representation prepared previously; 3. Selecting the sentences to be included in the summary (Allahyari *et al.,* 2017).
In the intermediate representation, the intermediate representation of the text is created to find the salient content based on this representation. A graphical representation of the text can be created. A graph is constructed involving all the sentences in the text documents. The text is summarised as a connected directed graph. The vertices are the sentences with a weight corresponding to the saliency/content score. The edges of the graph represent the weight of each sentence. The weight of each edge stands for the length of the original sentence before pre-processing. Broadly speaking, the intermediate representation could be achieved using 2 approaches: topic representation and indicator representation. The former scores a sentence based on how well it explains some of the most important topics covered by the text. The latter, however, computes the score by aggregating the evidence from different weighted indicators

2.8.1.2 Abstractive summarisation

This method uses more sophisticated NLP techniques to generate new summary entirely. This form of summarisation mimics the way humans generate summaries by 'understanding' the document and then generating a new bit of text of shorter form. An abstractive method is more complex as the methods try to replicate human intelligence to summarise documents (Lin, 2004). This also means that some part of the newly generated text which forms the summary may not even appear in the original text.

Most of the automatic summarisations methods (systems) are based on the extractive form of summarisation. Therefore, we would focus on this method of summarisation in our project. It might be better for researchers to see the original text. Abstractive will lose the original data. The key thing about extractive summarisation is to identify the relevant sentences from the document, put them together and present them as the summary. The first thing, therefore, is to identify the most important/relevant activities from the document. A human generated summary for the same text was generated for comparison with the automatic text summary

outputted by the machines to assess the efficiency of the automated summarisation approaches. Although the machine generated summary may not be the same as a human summary, the basic information contained in the two (2) summaries was compared. This is because the automatically generated summaries use *extractive* approaches while the human use *abstractive* approaches to summarisation. Factors such as readability, grammatical coherence and content would be used to evaluate the summary (Mani, 2001). The manual (human judgement) summarisation often requires many hours to accomplish. For example, a simple manual evaluation of a summary on a large scale over a few linguistic quality questions and content coverage as in the Document Understanding Conference (DUC) would require over 3,000 hours of human effort (Lin, 2004), hence automatic evaluation of the summaries would save time and energy.

A number of approaches have been proposed and proven to work effectively in identifying the relevant sentences. They are described in the subsections below.

2.8.2 Frequency-Based Approach

One of the approaches is the frequency-based approach. The frequency-based approach was pioneered by Luhn (1958). The work pioneered by this approach is to identify the relevant sentences using the word frequency. This approach has been implemented in two (2) ways: *word probability* and *Term frequency-inverse document frequency (TF-IDF)*. The work of Losad (2012) has demonstrated the use of this approach for phonological learning.

2.8.3 Feature-Based Approach

This approach identifies other features that determine the relevance of a sentence. This approach is termed *feature based*. He defined three (3) features that indicate the relevance of the sentence to be included in a summary. These features are sentence position, presence of title word and cue words. The work of Gupta and Lehal (2010), however, added other features such as: sentence length, term weight and proper noun.

2.8.4 Graph Based Approaches

The graph-based approach uses graph theory to address the summarisation task. In graph theory, objects and the connections between them are modelled using the following formula.

G = {V, E} where *V, E* stands for the vertex/node and the edge between the vertices respectively. In the context of text summarisation, the vertices are substituted for the sentences and the edges for the similarities between the sentences (i.e. the weight of the sentences). Using this approach, a document can be summarised by representing it as a graph and the sentences as vertices in the graph with weights of the sentences as the edges in the graph. The most widely used similarity measure is the *cosine similarity* (Erkan and Radev, 2004). This approach is an extractive approach to summarisation where important sentences are identified and then extracted for inclusion into the summary. From the constructed graph in this approach, important sentences are identified and extracted. Sentences which have connections to other sentences are considered relevant and, hence, extracted.

The two (2) well known algorithms to have implemented the graph-based approach are the *Hyperlink-Induced Topic Search (HITS)* algorithm (Kleinberg, 1998) and Google's *PageRank* algorithms (Brin and Page, 2012). LexRank (Erkan and Radev, 2004) and TextRank (Mihalcea and Tarau, 2004) are two (2) graph-based systems known to have successfully implemented these algorithms (HITS and PageRank).

2.8.4.1 TextRank for summarisation.

The TextRank technique is used to provide a comprehensive summary of documents using the idea of the TextRank algorithm. The TextRank is empirically adopted from the PageRank which ranks web pages in online search results. It measures the relative importance of each page based on the graph of the web (Page *et al.*, 1999).

2.8.4.2 PageRank

The PageRank algorithm inspired the TextRank algorithm and, hence, the idea is the same for both algorithms. As shown in the above section, the PageRank deals with pages and their rankings. The TextRank, however, uses sentences instead of web pages. The similarities between the sentences are the equivalent of web pages navigation, and such similarity between the sentences is used to measure or score the sentences. A word appearing in two sentences is a link between them to measure the similarities between the 2 sentences. The higher-ranking sentences are selected and put together to generate the summary for the documents.

2.8.4.3 LexRank Algorithm

This is also a graph-based method which employs the cosine similarity function to measure the similarities between different sentences. It uses a predefined threshold to build the graph of the documents by creating edges between the sentences any time the similarity between the sentences rises above the threshold. Just like the TextRank algorithm, LexRank uses the PageRank-like approach to rank the sentences to be selected for inclusion into the summary.

2.8.4.4 Cosine Similarity

The cosine similarity is a measure of similarity between 2 non-zero vectors (2 documents over a vector space) that measures the cosine of the angle of similarity between them. As the name suggests, it only measures the orientation, not the magnitude of the 2 vectors.

The measure of the cosine similarity of the 2 vectors is calculated using the Euclidian dot product as follows:

$$\boldsymbol{A.B} = \big|\big|\boldsymbol{A}\big|\big|\big|\boldsymbol{B}\big|\big| \cos \boldsymbol{\theta} \tag{1}$$

Given the two (2) non-zero vectors A and B, the cosine similarity cosθ, is calculated using the dot product and magnitude as shown below:

$$\boldsymbol{similarity} = \cos \boldsymbol{\theta} = \frac{\sum_{i=1}^{n} \boldsymbol{AiBi}}{\sqrt{\sum_{i=1}^{n} \boldsymbol{A2i}}\sqrt{\sum_{i=1}^{n} \boldsymbol{B2i}}} \tag{2}$$

The angle between two term frequency vectors cannot be greater than 90°. Hence, the resulting similarity ranges from −1 meaning exactly opposite, to 1 meaning exactly the same, with 0 indicating orthogonality or decorrelation, while in-between values indicate intermediate similarity or dissimilarity. Cosine similarity can be seen as a method of normalising document length during the comparison. For text matching, the attribute vectors *A* and *B* are usually the term frequency vectors of the documents. The vector space model, therefore, is the sentence matrix (TF-IDF scores).

2.8.5 Cluster-Based Method

Clustering refers to the grouping of similar instances into a cluster (Yogan *et al.*, 2016). In a cluster-based approach to summarisation, sentences are grouped into clusters of similar content. The similarity between sentences is computed using the *cluster centroid* method. High scoring sentences from each cluster are selected to form the summary for that document. Different

clusters may represent different sub-topics. K-means clustering algorithm is used to generate the clusters. This method of summarisation for multi-documents works by identifying the most central sentences from the documents which gives the basic and sufficient information about the central theme of the documents. The vector space model is a common way of identifying the central sentences. The centrality of any sentences is measured in terms of the centrality of its words. The term frequency inverse document frequency (TF-IDF) score is used to measure the centrality of the words. Words that have TF-IDF scores above a predefined cosine threshold are the centroids of the cluster.

The central idea for centroid based summarisation is that, different clusters are generated from the sentences in the documents. Sentences which have more words in common with the centroid of clusters are considered the central sentences i.e. sentences to be included in the summary. Hence, these sentences are put together and presented as the summary (Radev, Blair-Goldensohn and Zhang, 2001). The process is depicted in fig. 6 below. Sentence selection into any cluster is based on the similarity of the sentence to the topic or theme of the cluster C_i, the location of the sentence within the document L_i and the similarity of the sentence to the first sentence within the document of the sentence. The relationship can be represented by:

$$Si = W1 * Ci + W2 * Fi + W3 * Li \quad (3)$$

Where W1, W2 and W3 are weights of each cluster.

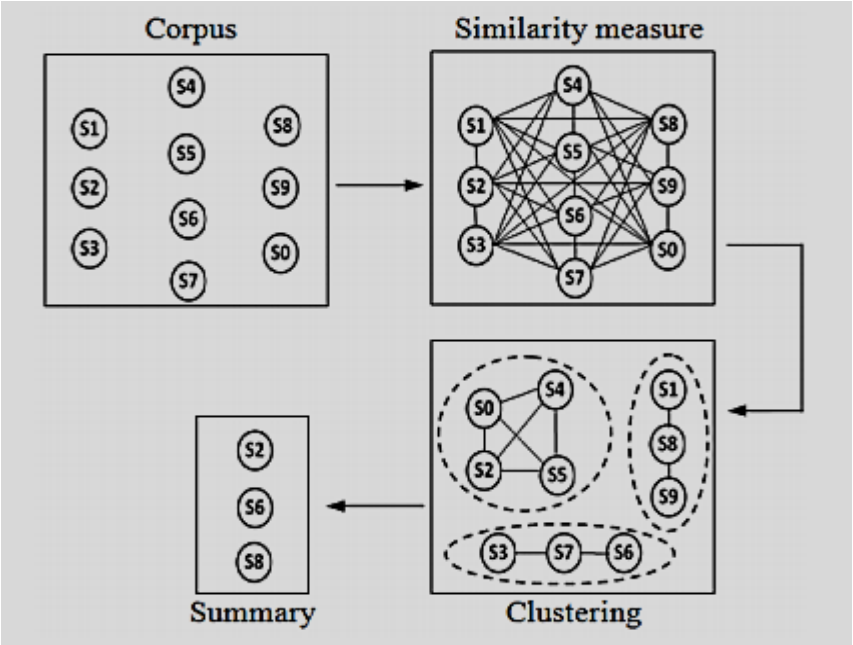

Fig. 2.1 cluster-based summarisation method

Since many clusters are generated from this approach with each cluster representing a different topic or theme contained in the document, this approach (centroid-based) is most suitable for document collection that contains different topics (Zhang and Li, 2009). However, a summary generated from this approach may not be representative of the document. This is because sentences are ranked only based on their similarity with cluster centroid which is simply frequently occurring terms.

Similarly, The existing research shows how machine learning based approaches can be applied for plagiarism detection (Al-Sallal *et al.,* 2019), flood management (Amin *et al.,* 2019), fault detection and isolation (Iqbal *et al.,* 2019), route optimization and self-learning vehicles (Birek *et al.,* 2018), relevance feedback for personal retrieval (Akuma and Iqbal 2018; Alhabashneh *et al.,* 2017; Iqbal *et al.,* 2017; Akuma *et al.,* 2016; Iqbal *et al.,* 2015; Grzywaczewski and Iqbal 2012, Grzywaczewski and Iqbal 2011; Grzywaczewski *et al.,* 2009).

2.9 Chapter Summary

This chapter explored, expressed and expatiated on the subject, the systematic review automation in software engineering and the other domains in general. The research gap was identified from the literature (see section 2.5). This provided the novelty for the research as well as the contribution to the area. Also surveyed, analysed and evaluated were the method, tools, and techniques relevant for the research.

3 Methodology and Experiment Design

In chapter two, the literature on the current state of science on the research problem was surveyed as well as the appropriate tools and approaches for supporting the systematic review automation in software engineering. In particular, the data extraction approaches and the degree of automation available were examined. The research gap was established indicating where this research effort fits into the bigger picture. In this chapter, the detailed methodology for the research is laid out. This chapter gives the general approach adopted in conducting the search, including the analysis of the problem, data collection and the experimental design. The chapter also captures, examines, and analyses the evaluations of the various parts of the system from the potential users of the system.

3.2 Methods

The main methods will be:

- Secondary research to establish the background to the domain and build on previous work.
- Prototype development using machine learning, natural language processing, text and data mining.
- Testing and user evaluation to assess the success of the research.

The researcher contends that:

(a) By using natural language processing and text mining, useful discriminatory content can be automatically extracted from papers and processed to a form that will support SLR.

(b) Given the results of (a), a tool can be built to search for relevant information and store it in a form that can be useful for future processing and enable interactive analysis.

3.3 General Approach to the Research

The general approach used in the research was quantitative and includes the experimentation, inference and then evaluations of the various phases. As detailed in section 1.10 in chapter one, the overall task is divided into three (3) phases as shown in the fig. 3.0 below.

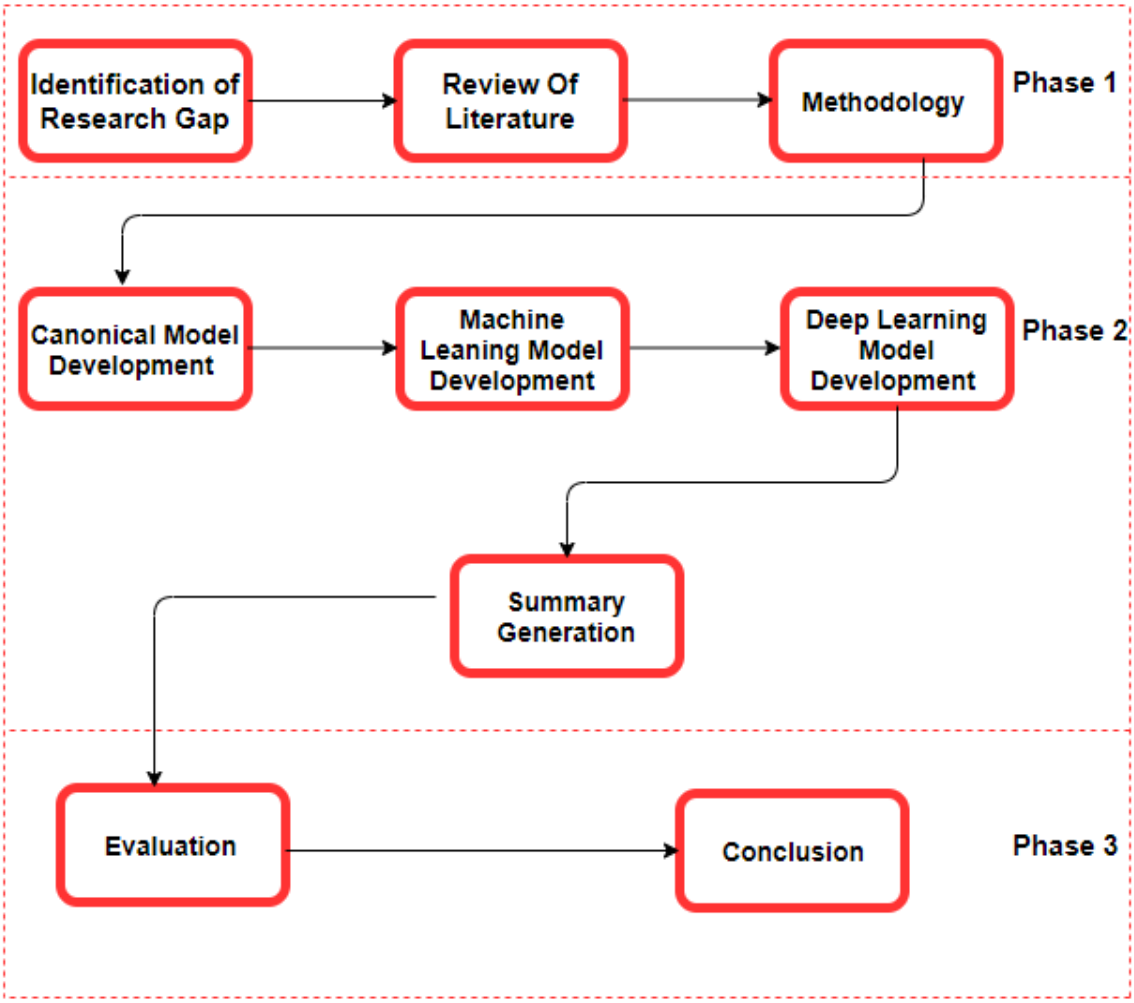

Fig. 3.1 Research Approach

However, in a more refined manner, the following steps were followed through the research.

Phase 1: In this phase, the relevant or related works or research efforts in the subject area were surveyed and the academic ground (a novel gap) for the research project was established. The literature was reviewed periodically to catch any relevant but missing or new research. Having established the grounds for the research, a research plan was set out. It involved the selection and appraisal methods and techniques needed for the project. The research design was also developed in this phase.

Phase 2: This phase consisted of four steps: (1) development of the canonical model; (2) development of machine learning algorithms to identify scientific paper sections; (3) improvement of the approach through application deep learning techniques; and (4) development of summarisation methods.

Phase 3: This is the final phase of the research. Here, various parts of the system were assessed, and the various components of the system were integrated. Potential users of the system performed the final evaluation and gave their feedback and assessment.
Finally, the conclusions and recommendations, including future work, arising from the research were considered and documented.

3.4 Canonical Model

As shown in fig. 3.1, the canonical model is part of the second phase of the research. The research is aimed at fetching the desired data from the primary studies (published articles/papers) to answer the systematic review questions. These research papers are well structured, well formatted and logically connected. Details of the implementation and the analysis that led to the canonical model are contained in chapter four (4). The approach taken to develop the canonical model involved the following stages: data collection, algorithm design, experiment design; building the canonical model and evaluation. A data sample of one thousand (1000) documents was used for the canonical model development. Using the algorithm developed for this purpose, the structure of the scientific papers was examined. The data (papers) were obtained carefully. This was to enable the study of the structure of the scientific papers in order to analyse and propose a structure based on the papers. Afterwards, a statistical analysis was performed, and the final structure called, *a canonical model* was arrived at. However, since the machines do not have an 'understanding' of the canonical model, machine learning models were developed to enable the realisation of the canonical structure.

3.5 Machine learning

The choice of machine learning for use in any given task is not a straightforward decision. Several factors must be considered such as the nature of the task, the type and size of the data, the expected output, and available computational resources (Asthana, 2020). Therefore, the choice of an algorithm depends on a combination of factors. The following factors have been identified in the literature as the most relevant consideration in selecting machine learning for any task.

1. Nature of Task:

Machine learning (ML) tasks are either supervised, semi-supervised, unsupervised or reinforced. ML algorithms are designed around these categories of tasks. The supervised

category of ML is mostly used for classification and regression. The models developed can then be used for prediction. For example, a company can predict its sales at different times of the year using the data from previous years. The previous data is used as an input to generate a prediction (output) based on new inputs. The key feature of supervised ML is the target variable for prediction (Rajoub, 2020). Examples of supervised ML methods are: Regression methods, Classification methods, Ensemble methods (a mixture of machine learning techniques with machine learning tasks), Neural networks (NN) and Deep learning (DL). NN and DL, however, can also be used for supervised task as it has been used in this research work.

Unsupervised category of ML does not have any target variable to predict. One way it could be used is for pattern recognition. It evaluates the pattern in the data and then forms clusters of items that are similar. A supermarket could segment its products with similar characteristics without having to specify in advance which characteristic to use (Rokaha, Ghale and Gautam, 2018). Example of unsupervised machine learning methods are: Clustering Methods (Xiao *et al.*, 2019).

2. Nature of Data

Another factor to consider is the nature of data. It is vital to have a clear picture of the data, the problem to be modelled as well as the associated constraints (Rajat 2018). Basically, data is either structured or unstructured. Unstructured data is not organised in any data model. Structured data on the other hand is organised in a model, usually contained in rows and columns such as relational databases, CSVs etc. hence, it is easy to search and organise as well as map its elements into pre-defined fields (Marr, 2020). About 80% of available data is unstructured (Blumberg and Atre, 2017). The lack of structure makes the unstructured data much more difficult to manage and analyse. For structured data, regression algorithms are the most suitable. For unstructured data, however, classification algorithms are far better. Works such as Kumar, Dabas and Hooda (2018) and Ball *et al.* (2018) have applied classification algorithms on unstructured tasks.

3. The Expected Output

The expected output also plays a role in deciding the algorithm to use. For a task where a label or class is expected as output from the algorithm, classification/regression methods are used. For classification, it returns a class or label as an output. For example, classifying a spam email, this is mostly represented as either 0 or 1. This means that the data is classified as belonging to one class/label (spam) or another (not spam). For regression, it computes a continuous valued output and returns a numerical value(s) instead of label as output. The expected temperature

for the week ahead is a continuous valued output. For tasks, however, where other forms of outputs are expected rather than label, such as clusters, clustering algorithms are used. Clustering divides the data into a group or clusters with each cluster having similar characteristics. For example, a supermarket could group together items which are usually bought together by customers and, hence place them together on the same or adjacent shelves. *K*-means, *k*-modes and Hierarchical algorithms are the most popular clustering algorithms unsupervised learning. Rokaha, Ghale and Gautam (2018) used Hierarchical clustering to enhance the supermarket operations by clustering similar items customs buy, thereby reducing the shopping time. Using a cloud-based framework, Shakeel, Baskar and Dhulipala used clustering algorithm to diagnose the diabetes mellitus by clustering the different age groups and gender. Xiao *et al.* (2019) have used *k*-mode to cluster to optimise Integer Linear Programming (ILP) and proved very successful. Other works that have utilised the clustering algorithms for similar tasks include Rappoport and Shamir (2018) and Rodriguez *et al.* (2019). Considering the above factors, Ashthana (2020) developed the framework shown in fig. 3.1 below, as the guide for selecting the appropriate machine learning method for use in different situations.

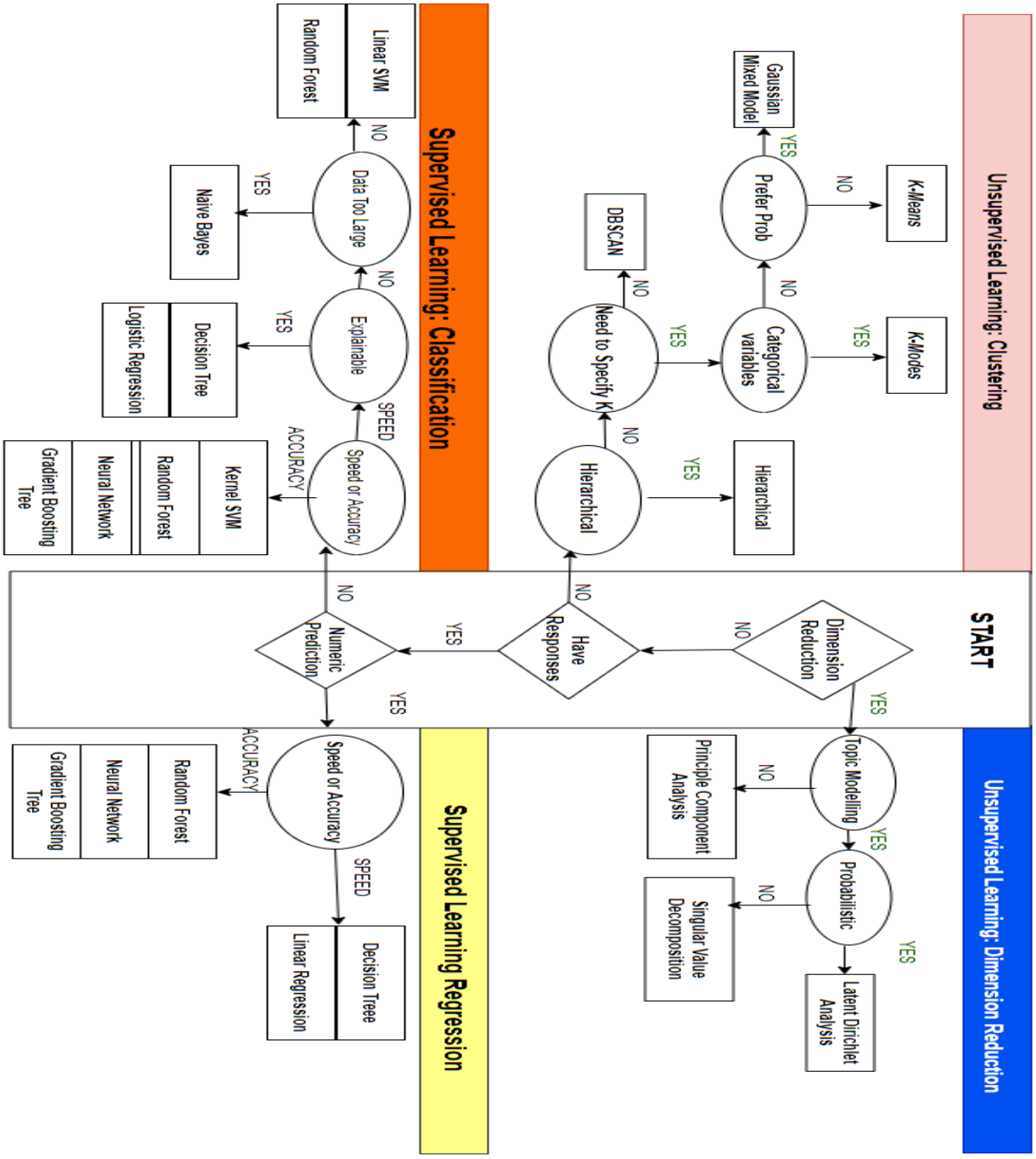

Fig. 3.2 Framework for selecting appropriate machine learning method

3.5.1 Selecting ML Algorithms for Information Extraction.

This research aims to build a model that would predict a chunk of text as belonging to one of six classes (output). We have a training data and target variable for prediction. That is, building an intelligent model that would identify the various sections of a scientific research document. We have a training and validation data sourced from the scientific research documents. The data is text heavy and not organised in any predefined model, hence is unstructured. The following points describe our tasks/processes. We would train the model to identify the various sections from the document.

1. We have training and validation data; hence the task is supervised.
2. The data is a free text from the SRDs, not contained in databases, excel sheet. Hence, it is an unstructured data.
3. The expected output from each identification is a label or category. For example, the model should identify/output the category or label such as a '*methodology', 'result',* etc.
4. Classification technique is used when the expected output variable is a category or label rather than a real or continuous valued variable (Harrington, 2012).
5. There are six (6) classes (sections) to be identified/classified.
6. There is large training and validation data. Thus, the need to strike a balance between the speed and accuracy. However, accuracy comes first.
7. Since it involves six (6) classes, the task is a multi-class task.

Thus, the task is a supervised multi-class classification task involving unstructured data aimed at training/building a model for the automatic recognition/identification of six sections from the SRDs. Clearly, the task is supervised, hence the supervised ML algorithms were selected as the appropriate choice for our task. Based on the various works reported in the literature, as well as the guide framework shown in fig. 3.2 (the framework itself is arrived at based on experimentation as well as review of the various works on the algorithms), the possible candidate algorithms for task are: Logistic Regression, Support Vector Machine (SVM) and Random Forest. Hence, these are the algorithms we selected for use in this work.

3.5.2 Support Vector Machine (SVM)

This is a discriminative supervised machine learning algorithm suitable for classification of both linear and non-linear data (Cortes and Vapnik, 2013). It is also used efficiently for regression tasks (Ahuja and Yadav, 2012). The main purpose of the SVM is to find the optimal

hyper plane (a decision boundary) that can be used to classify the documents into the respective classes). SVM is traditionally a binary classification algorithm, but several techniques have been adopted to make it work for a multi-class classification task. A common technique in practice has been to build one-versus-rest classifiers, commonly referred to as one-versus-all (OVA) (Schütze, Manning and Raghavan, 2008). However, in this research, another elegant multi-class SVM technique was used, a two-class classifier over a feature vector Φ ($\overline{x}$,y) derived from the pairs of the features and the class of the documents. During the validation process, the classifier chooses a class with a maximum score. The data consisted of 100% text, which is often linearly separable (Joachims, 1998) and has a lot of features, hence, the SVM was trained with a linear kernel which in turn makes the classification task faster. The implementation was achieved in Python and the popular library for machine learning algorithms known as the Scikit-Learn. The LibSVM (SVC) implementation of the algorithm was used, keeping the default values for the regularisation parameter of the error term, C, the kernel and the Gamma function γ. Fig. 3.3 below shows the separation hyperplane for the SVM. It separates the two (2) classes using the best separation line possible.

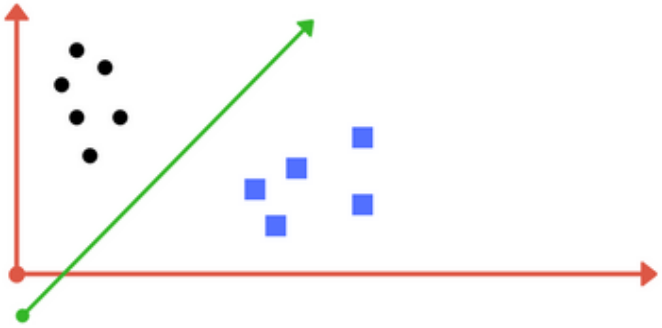

Fig. 3.3 the SVM hyperplane

3.5.3 Random Forest

This technique builds an ensemble of classifiers (multiple models of the same type) from different samples of the training dataset. Random Forest is a bagging algorithm which is also an extension of the decision tree. It builds models by drawing samples from the training data with replacement (Brownlee, 2016). It then aggregates the votes from different decision trees to decide the final class of the test instance. This ensures that the noise present in one decision tree is nullified through the votes from the other respective trees. The trees were constructed by considering a random subset of the features. Hence, the correlation between individual classifiers is greatly reduced. The default values for the *n_estimators* (number of trees in the

forest) and min_samples_split (minimum number of working set size at node required to split) were used.

3.5.4 Logistic Regression

This is a binary classifier that measures the relationship between the categorical dependent variable and one or more independent variables by estimating probabilities using a logistic/sigmoid function given in equation (3) below.

$$Q(z) = \frac{1}{1+e^{-1}} \quad (4)$$

All the above algorithms are linear classifiers. We chose the linear classifiers because the number of classes in our experiment is relatively small. So, our linear classification would be computationally inexpensive compared to when the number of classes is big. The computational complexity is ***O(kh)*** where *k* is the number of classes and *h* is the feature dimension of the dataset, in our case, the feature dimension of the text.

We implemented our task as a classification problem. The purpose of this activity was to train a machine to recognise the various parts of a paper according to the structure represented by the canonical model. The details of this implementation and results are reported in chapter 5.

3.6 Deep Learning

Deep learning algorithms such as the CNN are more sophisticated than the traditional machine learning. This is because they try to execute the task in the same way the human brain solves problems. Hence, the same task previously implemented with the traditional machine learning techniques was re-examined. The classification techniques were applied to the problem of identifying parts of a paper using CNN. The results of the CNN experimentation are reported in Chapter 6.

3.7 Summarisation

The last task in phase two of the project is the summarisation. The canonical model discussed in section 3.3 above depicts a structure that is intended to be used to enable the automatic extraction of the relevant information for the onward analysis and analytics in the review process. The machine learning experiments, including the deep learning, identified the relevant or intended sections which contain the needed or relevant data. The identified section is then

summarised to strip unwanted ‘noise’ or irrelevant information from the data. The best summarisation methods were selected following experimentation and evaluation.

Chapter seven (7) contains the details of the summarisation implementation including the justification for the various assumptions, choices as well as evaluations of the automatically generated output. The summarisation output was evaluated in two (2) ways. First using the ROUGE framework and second, by human experts.

The ROUGE stands for Recall-Oriented Understudy for Gisting Evaluation. The tool’s main function is to automatically determine the quality of a summary by comparing it to the summaries generated by humans (Lin, 2004). Variant forms of the ROUGE exist, including ROUGE-N, ROUGE-L, and ROUGE-W.

ROUGE-N is the n-gram recall between a candidate summary and the reference summary. The ROUGE-L is the longest common subsequence between the candidate summary and the reference summary. ROUGE-W is the weight longest common subsequence between the candidate summary and the reference summary. The details of choice of the ROUGE and implementation is all contained in chapter six (6).

3.8 User Evaluations

The third, and final, phase of the research was the user evaluation of the various components of the project, as well as the development of the overall conclusion. Potential users, of the system, were recruited for the evaluation of the various parts of the system. The evaluation was done in two (2) ways: evaluations using forms and the interview.

PhD students and staff at Coventry University who have conducted the review activity at professional levels were recruited for the evaluation. They signed a consent form at the start of the evaluation and then filled in the response form at the end of the evaluation. These forms are attached at the end of this thesis. Thirty (30) users were recruited for the evaluations.

They evaluated the overall usefulness of the model with respect to data extraction. Next, they made use of the machine learning models to identify some targeted sections. In the end, the summarisation methods were applied to the identified sections to give precise and concise results and the participants reviewed them. Then the evaluation was complete. The interview was conducted on three (3) experts who have years of experience on the subject.

3.9 The Dataset

The research focuses on a systematic review of literature in the software engineering domain. A dataset of six thousand (6,000) full text documents was collected. A subset of the dataset (One 1000 papers) was used to develop the canonical model. The full dataset was used for machine learning and summarisation experimentation.

3.9.1 Data Sources

The data for the training was collected from the SE SRDs. A broad view was taken on what constitutes a software engineering topic including innovative software design, creation and use. The choice of SE SRPs has been provided in section 1.2.

The relevant sources of documents for the research were identified from the literature. Sources such as IEEEXplore, ACM Digital Library, Science Direct and Springer were identified as the relevant data sources for the research. These sources have a high reputation for good research papers for both journal and conference papers in the software engineering domain (Muhammad *et al.,* 2018; Marshall, 2014).

However, this is just a tip for a start. The first paper was selected after an exhaustive evaluation by the researcher. The research has the background knowledge of the domain from which the documents were sourced, the software engineering domain. After that, a snowballing technique was used to get the related papers. Each paper has been assessed by the team before adding it to the collection.

Often, the journal papers contain more detail than the conference papers. However, useful information is also available in conference papers. Hence, both conference and journal papers were used. This is to ensure we all the relevant data available in the subject matter is captured and with as much detail as possible.

Using binary search, the six thousand (6000) papers in total were collected on the following software engineering topics: software design, software testing, software cost estimation, system analysis, machine learning, natural language processing, software cost estimation etc.

3.9.2 Search String Generation

Queries were composed corresponding to the topics for the data collection. Usually, multiple documents are returned for each query. Adding some sophistication could ensure that the

searches are narrowed down. This is done using the binary search string, and involves using logical operators such as ***OR, AND, NOT, quotation marks (“”) and brackets () etc.*** to connect the terms in the search query. So, for each of the following topics, we used the following search strings across all the selected sources (databases).

- *(Software OR tool OR system) AND Design*
- (*Software OR tool OR system) AND Testing*
- (*Software OR tool OR system) AND (“Cost evaluation” OR “Cost estimation”)*
- (*Software OR tool OR system) AND Analysis*
- (*“Machine Learning” OR “Artificial Intelligence”) AND Applications*
- (“*Natural Language Processing” OR NLP OR “Text Analytics”)*
- (*Biometrics OR fingerprints OR forensics) AND (Systems OR Approaches OR tools)*

3.9.3 Data Preparation

The data (papers) obtained are in PDF format. They were downloaded from these online databases and stored locally in a folder. Ideally, the PDF is a format for content presentation and not for text processing (Adobe 2018). The first task, therefore, is to be able to read the text and the extract the relevant text from these PDF documents. PDFMiner library was chosen for reading the text. The choice was because PDFMiner focuses entirely on extracting and analysing the textual data. It also allows the extraction of other important text attributes such as locations of text within a page, font style and lines. It also performs better in comparison with other similar tools (Pollock, 2016). The text files corresponding to the data were saved locally. The fig. 3.4 below shows the data collection process.

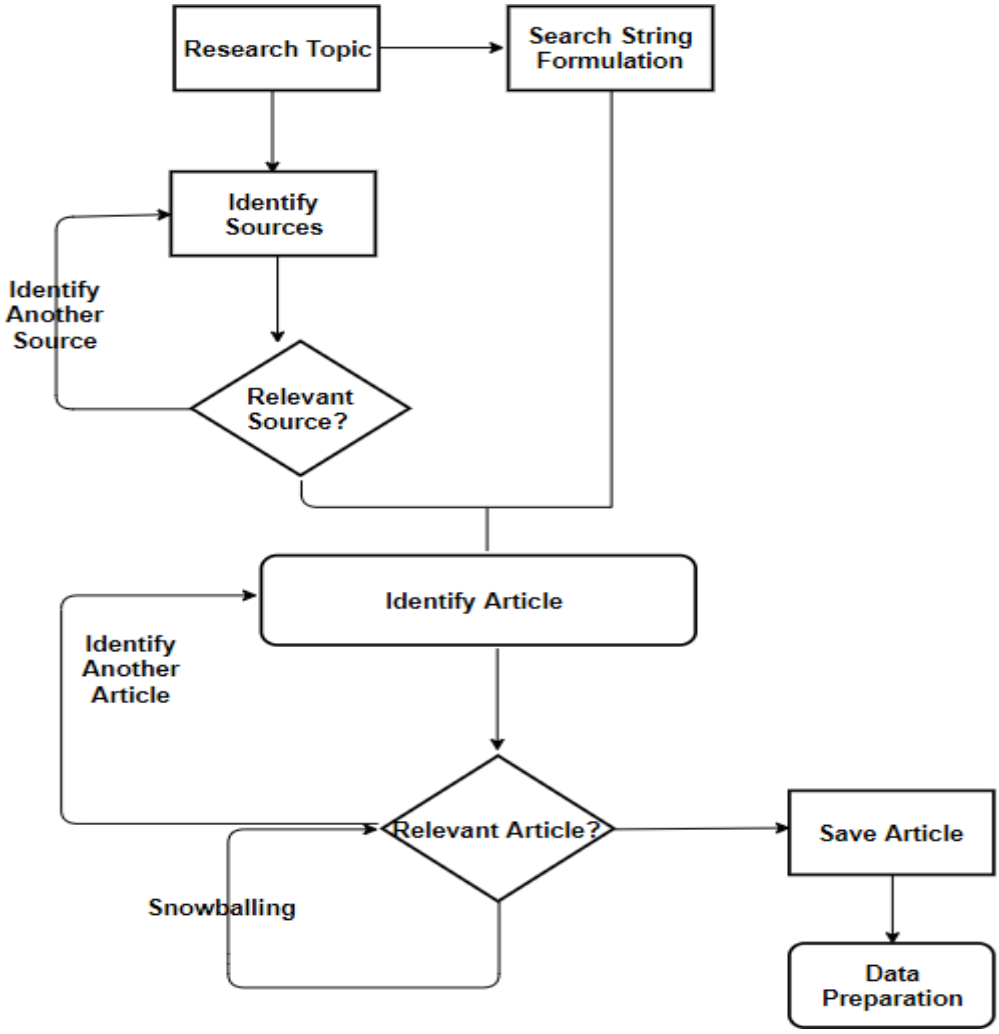

Fig. 3.4 Data collection process

3.10 Chapter Summary

The methods and the procedure for carrying out the overall research have been outlined. The chapter also specified the order and sequence of the respective tasks. Further details of the implementation and results are provided in the following chapters. The next chapter details the first task of the second phase of the research: the canonical model development.

4 The Canonical Model Development

The last chapter laid out the research methodology and the research design for accomplishing the research objectives. In this chapter, the details of the development of the canonical model is given. The canonical model is the intended framework to adopt for data extraction. The canonical model was developed, evaluated and adopted in this research.

4.1 The Canonical Model

The canonical model is a modelled structure of the scientific research documents (publications). The structure of the documents is well known to researchers, authors and students. As identified in the research problem, the goal of the research is to produce a novel framework suitable for extracting the relevant information from the SRDs.

The scientific research publications (literature) are organised in a structured format commonly referred to as IMRAD or IMRaD (Introduction, Methods, Result and Discussion & conclusion. IMRaD is the standard structure for the body of a scientific manuscript, after the title and abstract (Springer, 2020). This standard structure ensures that authors know what content should be included in each section of the document and ensures a logical flow of content. It also provides a map which readers can quickly follow to locate the content of interest in the manuscript. That should be at each level in the structure. It is envisaged that authors should use the different sections of the manuscript to "tell a story" about their research and its implications (Cooper, 2015; Springer, 2020). During a review, reviewers follow this (IMRaD) structure to read or appraise the content of a paper. Where needed, relevant data is also extracted from the various (IMRaD) logically connected sections of the paper. Because of the unstructured nature, as well as the volume of the papers involved in a review process, computer support (automation) has been identified as the key factor to improve the process as well as reduce the time spent in conducting the review and, hence, improves the accuracy.

In this research, we called the IMRaD structure as *canonical model of structure*. A model is a conceptual representation of a system made of the composition of concepts to help understand or simulate the object the model represents (Van Bakel, 2019). The canonical structure depicted in this research is a refined version of the IMRaD structure that involves sections as well as subsections of the IMRaD structure. It was arrived at after a structural analysis of the papers. This research built the algorithm used for the structural analysis. The structure is hierarchical involving sections and subsections ((Bates College 2011).

After having the conceptual canonical model in place, the next task is to 'teach' the machines (computer systems) to recognise this canonical (IMRaD) model through machine learning process. So, the purpose of this model is to identify and recognise all the IMRaD sections in a scientific research paper. We can then target a relevant section from this structure and then

extract the needed data from it using the summarisations methods presented in the later part of this thesis.

The point is, instead of readers to follow this standard structure to locate content (extract data) of interest, we allowed the machines (computer system) to take over and execute this task. Hence, the IMRaD structure would first be understood and recognised by machines (computer systems). Using this structure, the target section is identified, and then automatic summarisation is applied to summarise the content and present the result to the user. The result is the relevant data/information from the publications. The experiment is described in the subsections below.

4.2 Experimental Procedure

The approach taken to develop the canonical model involved an experiment with the following stages: data sampling and preparation; algorithm design; experiment design; building the canonical model and evaluation. The data sample is meant to explore how the various section headings are reported in the papers such that the heading titles or concepts is used for the intended task. The algorithm is for the identification of the various headings/subheadings in the various sections of the papers. The identified section headings names (terms or words or phrase) would be used for the analysis and evaluation to select the final candidates for the canonical model. These stages are described in the following subsections below.

4.3 Data Preparation and Sampling

For the canonical experiment, a sample data was taken from the data pool for the project. A sample of one thousand (1000) published full-text academic research papers was used. The papers, in PDF format, were stored locally and in separate folders. The PDF format is for the presentation of content not for text processing (Adobe 2018). Therefore, the first task is to be able to read/extract the relevant text from the PDF documents. PDFMiner library was used for reading the text from the PDF. PDFMiner was used because it focuses entirely on extracting and analysing the textual data. It also allows the extraction of other important text attributes such as locations of text within a page, font style and lines. This performed better in comparison with other similar tools (Pollock, 2019).

4.4 Algorithm Design

For scientific research articles, every section is reported under a named heading. However, to the best of the researcher's knowledge, there is currently no computational procedure for the identification/extraction of various section headings from full-text documents (scientific research articles/papers). This research proposes a novel approach for that purpose. The basic structure and formatting properties in the papers were explored and, hence, the algorithm was designed after a careful analysis of the documents (papers). The documents used in the experiment consist of different two (2) document formats: PDF and Docx, each converted to raw text (.txt) but retaining the original formatting. The algorithm is rule-based, unsupervised and natural language based, and is detailed below.

Algorithm

---

1. Pull out the entire texts from the PDF/Docx document.
2. Divide the extracted texts into paragraphs (sections).
    (a.) Use double line spacing (\n\n) to identify and divide the text into sections. If not, go to (2b.)
    (b.) Apply single line spacing (\n). If not go to (2c.)
    (c.) Apply sentence tokenization.
3. Pick out sections that begin with numbers (either Arabic or Roman). If yes, go to (5).
4. Break the entire text into sentences using sentence tokenization. Go to (7)
5. Process the texts.
    (a.) Get the length of the first part of the text. If length <50 then go to 5(b.) else 5(c.)
    (b). Check the number of special symbols. If number >3 then go 5(c.). Else go to (8).
    (c.) Get the next text. Go to 3.
6. Analyse the text font style
7. Extract and store the headings.
8. End.

The fig. 4.1 below shows the flowchart of the algorithm. The sections in most papers are numbered. The numbers are either in Roman or Arabic. The algorithm first looks for this property (in addition to other properties) to identify the section headings. Where the sections are not numbered, only font style attribute is analysed. The font style may be bold, capitalised italic, underlined or a combination of these. This adds to the efficiency of the approach. A critical challenge is when a chunk of text or section (not a heading or subheading) has this same feature i.e. is preceded by an Arabic or roman number. To cover that case, the algorithm incorporated further features in order to make the identification, and this includes special character feature. Most headings do not contain special characters (e.g. £, $, %, ^, > etc.). So, a special feature was added to the approach. Also, the length of the text is another criterion for section header identification. The headings contain few words as they are but the description

of the section. This makes the approach to incorporate the 'text length' as a criteria to identify the headings. A text length of 50 is used to distinguish between headings and non-headings. Most section headings are short. In the sampled data, most were shorter than 50 characters. There are, however, others with longer text sizes.

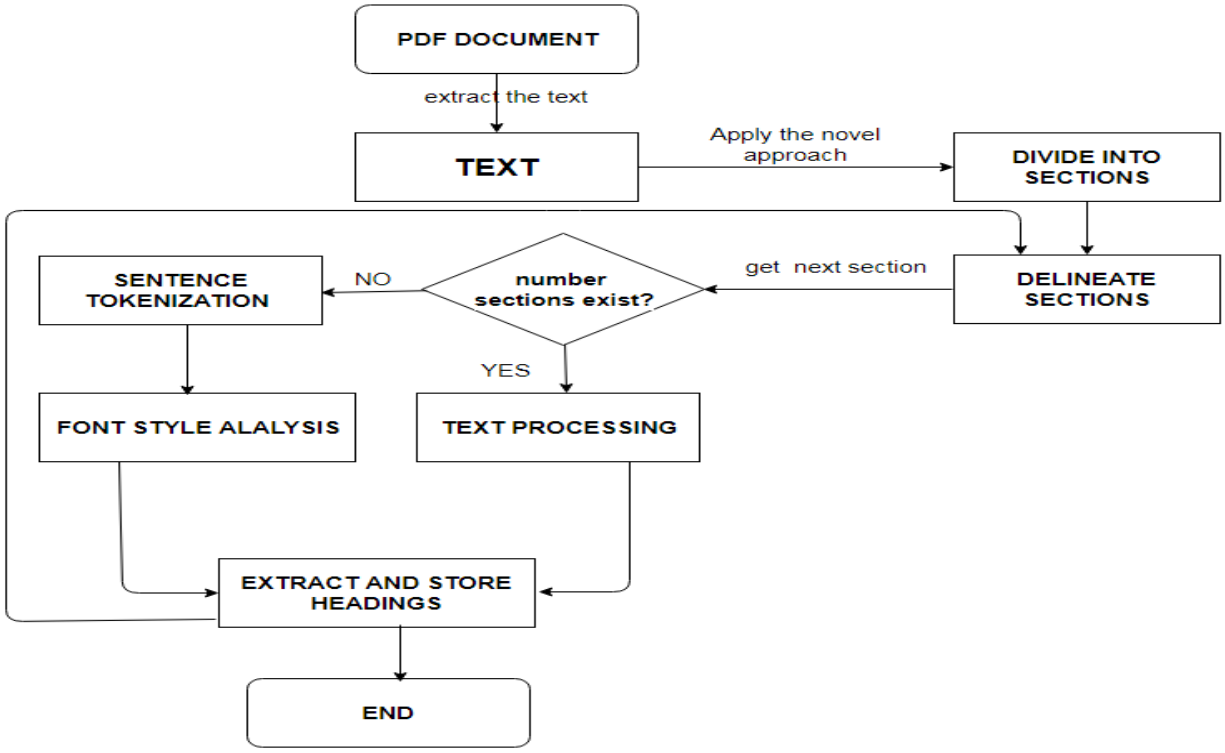

Fig. 4.1 The flow chart of the algorithm.

Fig. 4.2 below shows a sample output generated by our novel approach/algorithm. It identifies the headings and the associated text from most of the documents. From the 1000 documents used for the experiment, the algorithm correctly picked out the section headings and their associated texts in 820 documents. Due to the fact that research articles have different number sections, the following formulae was used to calculate the success rate of the algorithm on any given document.

$$X = \frac{\textit{Number of section headings identified}}{\textit{Total number of sections in the document}} \qquad (5)$$

Where X represent the success threshold of the algorithm on any document, the value of X was approximated to the nearest whole number. 1 indicates success and 0 otherwise. For example, a document with 50% or higher identification for all the sections contained in it, was scored 1

while those with less than 50% were scored 0, meaning the algorithm was not effective on such document.

Applying the formulae to the result of our experiment, we obtained the success rate of the algorithm is calculated as follows.

$$Accuracy = \frac{number\ of\ documents\ identified\ correctly}{Total\ number\ of\ documents\ in\ the\ collection} \times 100 \quad (6)$$

$$= \frac{820}{1000} \times 100$$

$$= 82\%$$

Therefore, the algorithm has a success rate of 82%. Meaning that out of every 10 documents presented to it, it can detect and extract the majority of the section headings in at least 8 documents. Put in another way, a document with, say 10 sections, if the algorithm can identify the majority of the sections (at least 6), the algorithm is successful on that document. From the experiment accuracy, out of every 10 documents, the algorithm succeeds on at least 8. This novel approach led to the extraction of headings from most of the PDF documents for further processing. The fig. 4.2 below shows the sample output from the algorithm. These are the extracted headings from the papers using the approach.

Paper Id	Section One	Section Two	Section Three	Section Four	Setion Five	Section Six
1.pdf	I. INTRODUCTION	II. REQUIREMENTS TOWARDS ENTERPRISE MODELING	III. COMPARISON OF ENTERPRISE MODELLING	IV. TAKE HOME POINTS	V. RELATED WORK	VI. CONCLUSION AND FUTURE WORK
10.pdf	I. INTRODUCTION	II. SESRA TOOL	III. DISCUSSION	IV. CONCLUSIONS		
100.pdf	I. INTRODUCTION	II. MODEL	III. MAIN RESULTS	IV. PIR SCHEME CONSTRUCTION FOR b = 1	V. PROOF OF THEOREM 1	VI. CONCLUSION
1000.pdf	I. INTRODUCTION	II. RELATED WORK	III. EXISTING SYSTEM	IV. PROPOSED SYSTEM	V. METHODOLOGY	VI. IMPLEMENTATION

Fig. 4.2 Sample output generated from the algorithms.

4.5 Implementation

We implemented the above described algorithm on all the sampled data. A ‘tool’, developed in Python, was specifically developed and used for this experiment. Python’s NLTK module provides all the necessary support for the NLP tasks. This is appropriate for our task. The tool employed the novel algorithm (proposed by this study) on our sampled data for the identification and extraction of the various section headings and their associated text from all the PDF documents which were locally stored in a folder. The identified headings were then taken to the next stage of the process, the stop words removal. This is followed by stemming and synonyms aggregation to build the corpus for the analysis of the more frequent terms/words/phrases contained in the corpus. Finally, the potential candidates for the canonical model were then selected using a threshold measure. This is shown in the fig. 4.3 below.

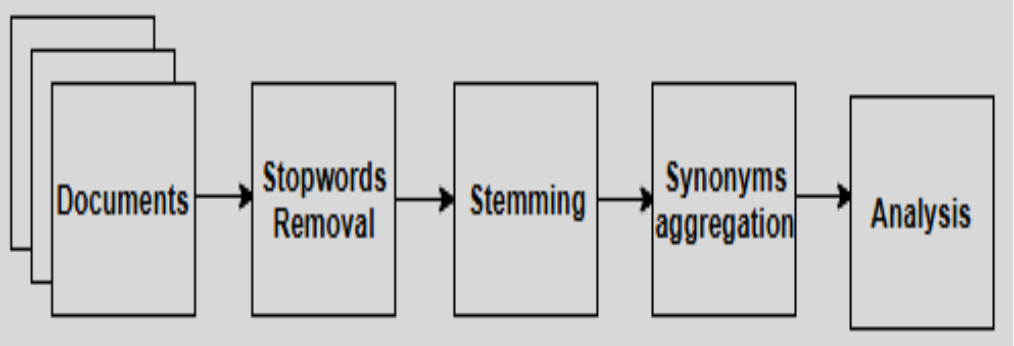

Fig. 4.3. The experiment flows

4.5.1 Stop Words Removal

The stop words are commonly and frequently occurring words such as ‘the’, ‘is’, ‘was’, ‘a’, ‘an’, ‘about’, ‘in’, ‘that’, ‘of’, ‘for’ etc. These words are considered as ‘noise’ in the data and, hence, do not add value to the text processing tasks. So, removing them would improve the accuracy of the process. Python has an in-built list of stop words. They were used as a reference to successfully remove the stop words from our data, leaving only the phrases/words/terms that have a high impact on text processing.

4.5.2 Stemming

Some words exist in many variations from the same root. For example, method, methodology, methodological etc. have the same root. As a normalisation method, stemming traces back every word to its root. This ensures that the word is captured irrespective of the variant form it was used. In our experiment, we ensured this normalisation method to smooth out the variations in our corpus. For example, words such as Results, Result etc. Python NLTK was successfully used to achieve this normalisation.

4.5.3 Synonyms aggregation

Several words can be used to describe the same the thing. In our experiment, we encountered terms such as Literature Review, Related Work and Previous Work, all referring to the existing works done upon which the paper builds. Similarly, Methodology and Approach referring to the procedure followed in conducting the research. To address this difference in the choice of words to report the same concept, WordNet (Fellbaum, 2010) was used for the synonym normalisation. WordNet is like the thesaurus such that it groups words based on their meaning. This ensures that same concepts are captured irrespective of how (the choice of words) they were reported.

4.5.4 Frequency Analysis

After the stages in 4.5.1, 4.5.2 and 4.5.3 respectively, the corpus consisting of the most valuable terms/words/phrases is built. Note, however, that for stages 4.5.1 and 4.5.2, each text from respective headings is processed individually. Afterwards, the 4.5.3 stage was applied on the entire corpus (consisting of the entire text collections). A simple frequency tally of every term/word/phrase in the corpus was taken to select the topmost occurring terms as the representative candidates for the models using a threshold of the counter of 100. The 100-mark threshold was used as it represents 10% of the total number of documents used for the experiment. The frequency distribution of the words/terms/phrases in the collection is shown in the figure below.

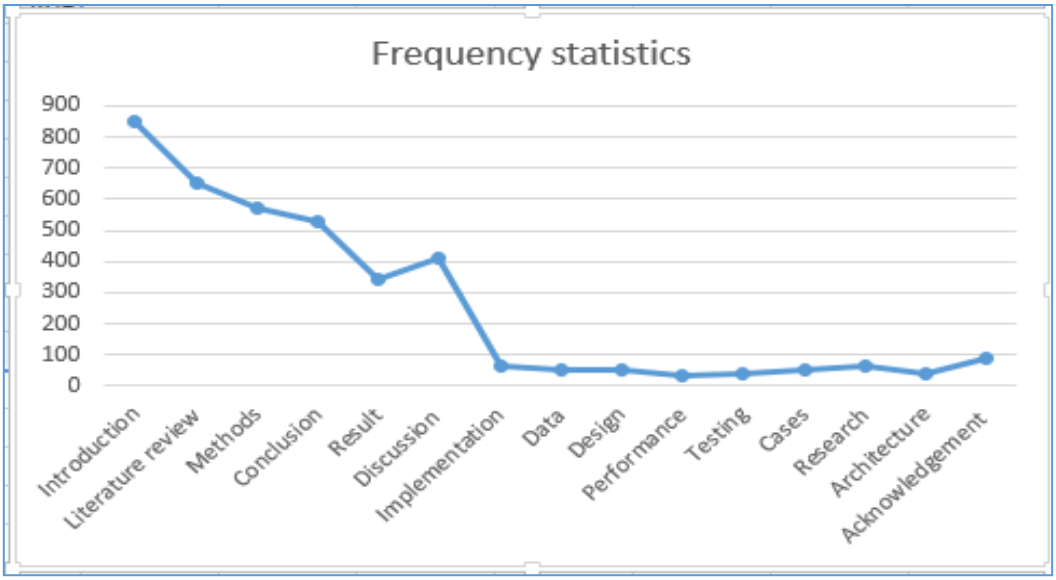

Fig. 4.4 Frequency Analysis

From the result of the frequency analysis and the tally computation of the candidates, 15 candidates had a tally or frequency of more than 50. But since we set a threshold of 100, only 6 candidates met that tally threshold of 100. Hence, 6 candidates were selected for the canonical model development.

From the frequency statistics shown in Fig. 4.4 above, six (6) candidates: *introduction, literature review, methodology, result, discussion and conclusion* met the threshold of 100 and so are included into the model. Fig. 4.5 below shows the final canonical model of the structure. Of equal significance is the fact that sections such as background, methodology and results

have one or more subsections associated with them, while the rest of the sections have no subsections at all. This is true for most of the papers.

4.6 The Canonical Model Generation

From the statistical analysis, six candidates met the requirement and thus, were selected for inclusion in the canonical model. Some of the candidates (sections) of the model, such as *methodology* and *results*, have subheadings (subsections). Others, however, such as *introduction, conclusion and discussion* have no subheading or subsections. The canonical model is shown in fig. 4.5 below. This model is not altogether new. The content of this model is common sections found in most journal and conference papers. What is of significance, however, is that this is the first time that such a model has been proposed for use in data extraction in SLR.

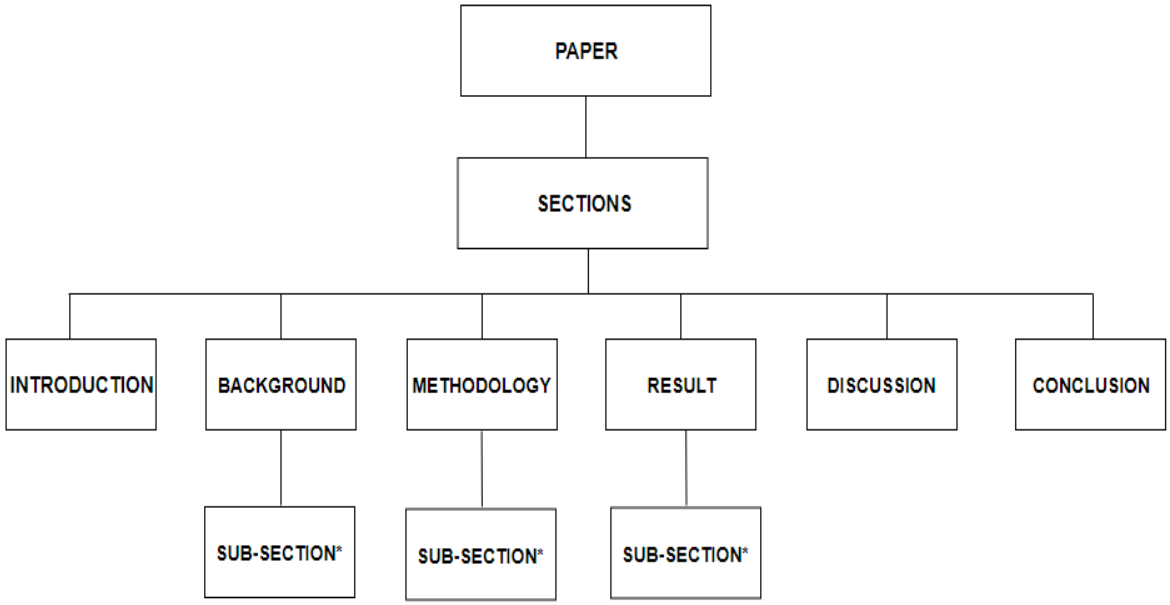

Fig. 4.5 The final canonical model

The intention is to match any paper to this model. The challenge, however, is machines do not have true 'understanding' of this model and so artificial intelligence procedures are created to enable the system "understand" the model in order to find and analyse the relevant contents required. Text and data mining including machine learning techniques will be used to obtain the results. However, not all the papers will match exactly to the canonical structure developed in this research. It will be part of the research challenge to develop a suitable machine learning

algorithm that can match any research paper to the canonical structure. The degrees of accuracy regarding the matching obtained from the system varied. However, appropriate techniques to deal with this task were also utilised. After matching the papers to the model, the desired sections can be delineated for the extraction and summarisation. Also, building the (machine) artificial intelligence to realisation of this model (canonical structure) is another research challenge of novel note.

4.7 Chapter Summary

This chapter highlighted the detailed approach followed to arrive at the canonical model, a unified approach adopted for the data extraction purposes in SLR of scientific research articles. This approach (canonical model) is the first of its kind. This is does not mean the idea of the structure is new, and it only indicated a new way to harness data from the papers. Having achieved the model, the machines need to be able to use the model. This means that we must teach the machines how to identify and use the contents in the model with respect to the scientific research papers. Hence, the next chapter explored the machine learning approaches and implemented the problem specified above. The next chapter reports the work on the machine learning implementation.

5 The Machine Leaning Models

In the last chapter, the canonical model was developed and reported. The model is a representation of the structure of scientific research documents (journal and conference inclusive). It depicts the various sections within the SRD. Machines, however, do not have any understanding of this structure and hence, the need to train them to do so. In this chapter, the machine learning experiment to realise this model is described. It is implemented as classification problem. The different machine learning algorithms are evaluated and the most appropriate for the task is selected. The task is to be able to identify the different sections in a SRD document.

5.1 Text Classification

The task here is to be able to implement the canonical model depicted in fig. 4.5, such that machines (computer systems) would be able to understand and use it. It involves the ability to identify and classify an unstructured text as belonging to one of the six (6) predefined classes. Hence, the problem was implemented as a text classification task. Since there are six (6) classes for the classification, a multi-class classification approach was used. A multi-class classification problem involves the assumption that an instance (a data sample) is assigned to one and only one class or label. For example, an instance from the data must correctly be classified into one and only one of the six labels for the machine learning task. The implementation details are fully described in the sections and subsections below. The machines were trained using a prepared and labelled training dataset. After the training, models were developed which we tested and validated using the test dataset. The algorithms learn from the training dataset; hence, it is a supervised machine learning process. Given input variables (x) and output variable (y), the algorithms learn the mapping function given by the equation below.

$$Y = f(X) \tag{7}$$

5.2 The Machine Learning Process

The machine learning process involves a number of stages. The number of stages varies depending on the type of learning. Since this is a supervised machine learning process, the learning process depicted in the fig. 5.1 below was applied. It involves data pre-processing, feature engineering (extraction), machine learning algorithm selection, training (function mapping), and machine learning model or classifier building and validation/testing of the model.

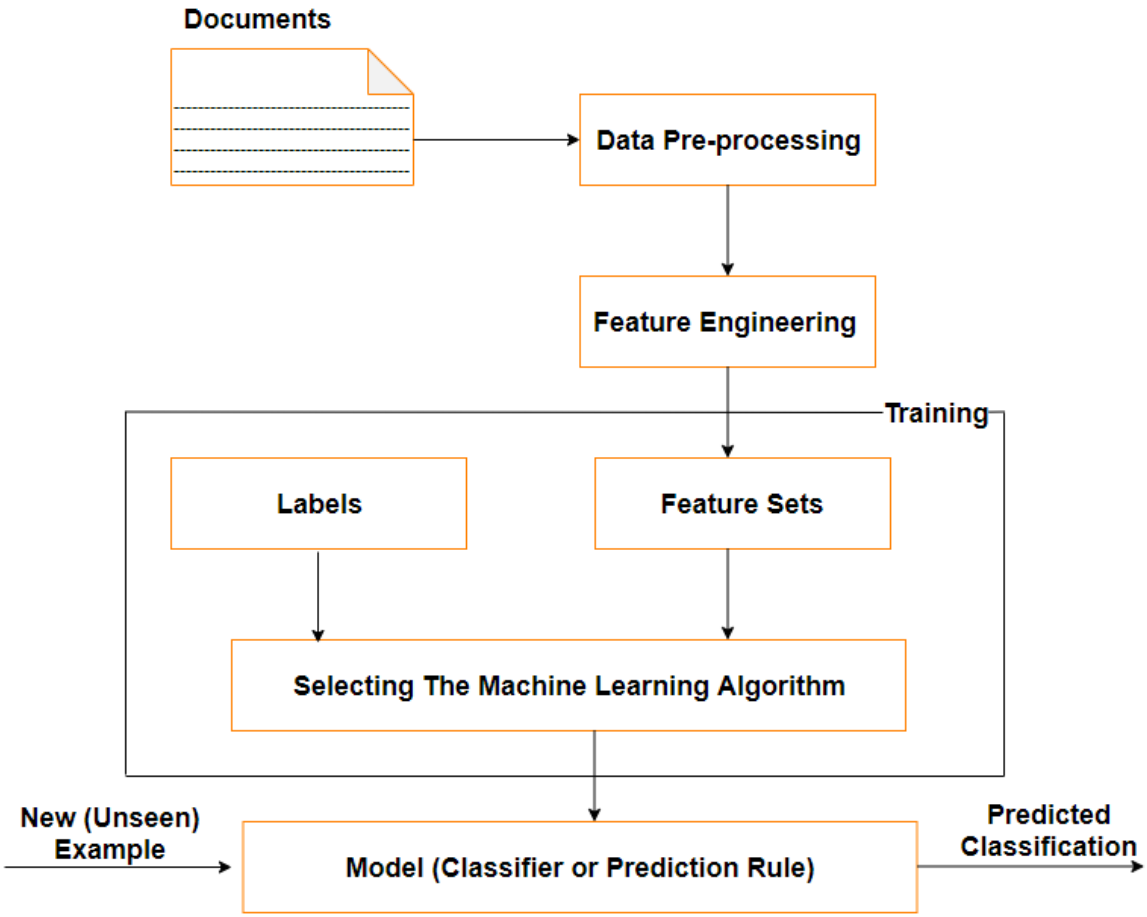

Fig. 5.1 the machine learning process.

5.3 Data Pre-processing

For machine learning in natural language processing, the data is mainly text, hence it requires pre-processing to make the data suitable for the machine learning experiment. It involves cleaning and transforming the data. In our experiment, the following pre-processing activities were performed on the data: tokenization, removing unnecessary tags, removing stop words, stemming and lemmatization.

5.3.1 Tokenization

This is the process of converting or breaking down the text into *tokens.* During this process, all unnecessary characters such as punctuations are thrown away. The tokenization is done at sentence level or word level via sentence and word tokenization respectively. In sentence tokenization, the corpus was chopped into sentences using the sentence splitter. For words tokenization, whitespaces are the delimiters for dividing the respective tokens. Consider the following snippet from the data where words and sentences were tokenised.

Input: "The result from the experiment shows that software maintenance cost is higher than the cost of initial development. Hence, software maintenance is very expensive". **Output:** The respective output for both sentence and word tokenization is as follows:

Sentence tokenization: this produced two sentence tokens:

- *The result from the experiment shows that software maintenance cost is higher than the cost of initial development.*
- *Hence, software maintenance is very expensive.*

Word tokens: The following word tokens are produced.

The, result, from, the, experiment, shows, that, software, maintenance, cost, is, higher, than, the cost, of, initial, development, hence, software, maintenance, is, very, expensive,

This step of the data pre-processing is very important because the feature engineering described in section 5.4 is based on the output from this stage.

Python has inbuilt punctuality for tokenization for both words and sentences.

5.3.2 Stop words removal

Our text classification task involves a special domain: the software engineering domain. Just like in other classification tasks, not every word impacted on the classification. Hence, the 'noise' (unimportant) words of data was removed. These were the stop-words. These are common and frequently occurring words such as '*the', 'is', 'was', 'a', 'an', 'about', 'in', 'that', 'of', 'for'* etc. These words are considered as 'noise' in the data and, hence, do not add value to the text classification and other tasks. So, removing them would improve the accuracy of the process. We used Python's inbuilt list of stop-words to remove them from our data. The cleaned text is then passed to the next stage of the machine learning process.

5.3.3 Stemming

This is the process of trimming down a term to its root word by removing inflexion. This is done by removing unnecessary characters, usually prefix. Stemming algorithms work by cutting off the end or the beginning of the word, taking into account a list of common prefixes and suffixes that can be found in an inflected word. This indiscriminate cutting can be

successful on some occasions, but not always, hence, this technique presents some limitations. Consider the following stemming example in table 5.1 below.

Table 5.1 the stemming

Form	Prefix	Stem
Studies	-es	Studi
Studying	-ing	Study

5.3.4 Lemmatization

Like the stemming, this approach removes inflexion by determining the part of speech and utilising a detailed database of the language. In other words, lemmatization considers the morphological analysis of the word. This means that it is necessary to have a dictionary (or a database) from the lemmatization algorithms can look through to link the word to its lemma. Consider the following example in table 5.2 below

Table 5.2 lemmatization

Form	Morphological information	Lemma
Studies	Present tense of the verb study, third person singular	Study
Studying	Gerund of the verb study	Study

5.4 Feature Engineering

Machine learning algorithms cannot work with the raw text directly, hence, it must be converted into numbers, specifically, the vector of numbers called vector space representation. Therefore, we performed the feature extraction/encoding by numerically representing the volume of text in the vocabulary. We used the bag-of-words model which is a simple but popular method of feature extraction or encoding. The bag-of-words model measured the occurrence of each unique word in a document using the vocabulary generated by all the words in the documents collection. In this approach, we looked at the vector space model of the words of the text, i.e. considering each word count as a feature. The bag of words model ignores both the grammar and order of the words in the document. The numerically converted text formed

the extracted features used for the experiment. The most relevant words from the stage above were then successfully used for the bag of words model that is then passed to the machine learning classifier. Each document from every class is mapped to the vocabulary, and the numerical representation is obtained from it. Table 5.3 below shows a sample of the bag-of-words model from three (3) documents: Doc1, Doc2 and Doc3.

Table. 5.3 The bag of words model

Terms	Approach	Classifier	Comparison	Indicate	Simulation	Process
Doc1	1	1	1	0	0	1
Doc2	0	0	1	1	0	1
Doc3	1	1	1	0	1	1

We trained our model on three (3) feature types:

- word count
- NGRAM and
- Term Frequency-Inverse Document Frequency (TF-IDF).

Feature engineering is a human craft rather a machine learning task. Choosing the right features often improves the performance of text classifier (Manning, Schütze and Raghavan, 2008). Hence, we chose the above feature types because they allow the extraction of the most descriptive terms from the document. Word count detects the vocabulary used (Forman, 2007), the NGRAM how the words are arranged (syntax) (Tripathy, Agrawal and Rath, 2016) and TF-IDF measures the importance of each word within the document (Wang *et al.*, 2017). Combining these features ensures classifier speed and effectiveness is enhanced (Zheng and Casari, 2018). Using a combination of these features is more accurate than using just one of them alone (Lin *et al.*, 2016).

5.4.1 Word Count

The word count is a simple tally for the presence or absence of a word. It follows the bag of words model described above and then a count of the number of times each word appears in the document. For example, consider the following text from two documents.

Document 1: The software cost estimation term

Document 2: NLP term is outstanding

The word count from our data is shown in table 5.4 below.

Table 5.4 Word count sample

Documen1		Document 2	
Term	count	Term	count
Software	*1*	*NLP*	*10*
Cost	*2*	*Outstanding*	*2*
Estimation	*5*	*Is*	*34*

5.4.2 N-Gram

Word count (Bag of words) does not take the order of the words into consideration. Hence, other additional measures were needed to reduce this independence of words. N-gram is an additional feature that captures the spatial information about the local word order (Wang and Manning, 2012). We implemented the N-gram feature using an N-gram range of (2, 3). This ensures a fast and memory-efficient feature mapping by using the top 20,000 features (N-Grams).

5.4.3 Term Frequency-Inverse Document Frequency (TF-IDF)

Term Frequency *(TF)* is the number of times a term appears in a given document. It is calculated as follows:

$$\textbf{Term Frequency (TF)} = \frac{\text{Number of times term t appears in a document}}{\text{Number of terms in the document}} \quad (8)$$

TF measures the importance of a word within a document by considering its frequency of occurrence within that document. It is thought that a frequently used term in the document indicates its importance in describing such document. However, term frequency is not sophisticated enough for adjusting the frequency of commonly used words (Silge and Robinson, 2018). By using the *Inverse Document Frequency (IDF)*, we ensured that the weight of the frequently used words is reduced and thus, increasing the weights of terms of the less frequently used in the collection of documents under consideration. The IDF is calculated using the formula shown in equation 8 below:

$$idf(term) = \log\left(\frac{N}{n}\right) \quad (9)$$

Where N is the number of documents and n is the number of documents containing the term (word). The IDF of a rare word is high, whereas the IDF of a frequent word is likely to be low. Thus, having the effect of highlighting unique words.

The Term Frequency-Inverse Document Frequency is calculated as the product of the term frequency and the inverse document frequency, as shown in equation 11 below.

$$TF - IDF = TF * IDF \quad (10)$$

Where:

TF (t) = (No. of times term t appears in a document) / (Total no. of terms in the document).

IDF (t) = log_e (Total number of documents / Number of documents with term t in it).

Hence, TF-IDF technique was employed because the classes are closely related with lots of common words which occur with a high frequency. Also, giving the high frequency of the noise (stop words), fewer (less frequent) words hold the deciding power to separate the classes in our collection. The result of using this technique improves the accuracy of the classification significantly.

Reflecting the TF-IDF on table 5 in section 5.3.1, the TF-IDF method heavily penalises the word ‘cost’ (0.6) but assigns greater weight to ‘estimation’ (1.5). This is due to the IDF part, which gives more weight to the distinct words. In other words, ‘estimation’ is an important word for Document1 from the context of the entire corpus.

5.5 Implementing the Machine Learning

We searched and conducted a preliminary investigation into the machine learning algorithms which are efficient in the text classification process. Our investigation led to the identification of the state-of-the-art machine learning algorithms with which we trained our classifiers for the outlined task. In the end, the results were compared, and the best preforming classifier was selected as our model. The algorithms used include: SVM, Random Forest and Logistic Regression. The procedure for the experiment is described in the section 5.2.

5.5.1 Support Vector Machines

The support vector machines (SVM) is a binary classification algorithm, fit for a 2-class problem. The SVM classifies the data by finding the optimal hyperplane that separates the two (2) classes. The optimal hyperplane is the hyperplane with the biggest margin between the classes. The fig. 5.2 shows the optimal hyperplane which separates first class 'A' and the second class 'B'. The boundary data items are the support vectors which are the borderlines between the classes.

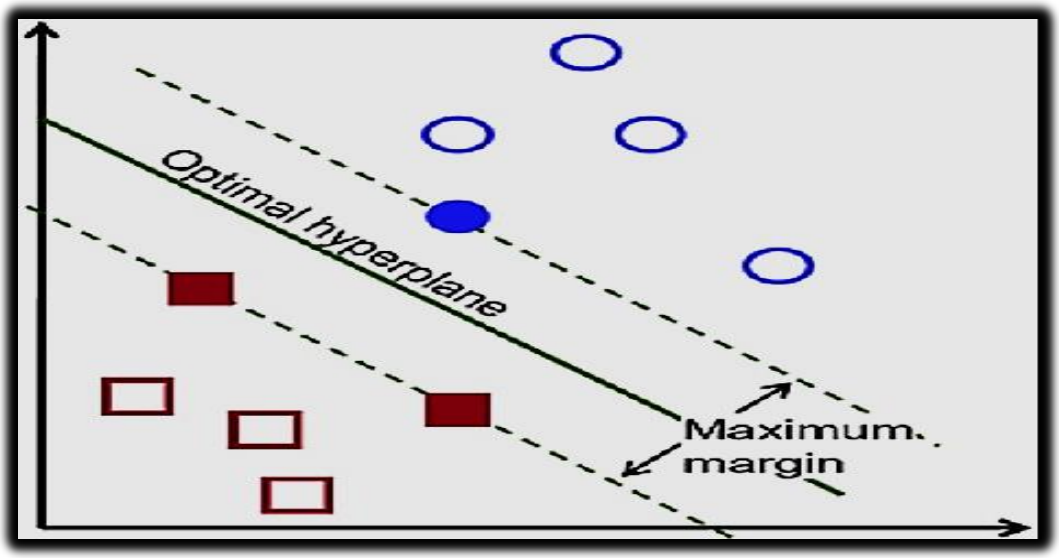

Fig. 5.2 the optimal hyperplane

To adopt the SVM for the multiclass problem, the *one versus all (OVA)* technique was used. This is also known as a *one against all (OAA)* technique. This way, one class is determined by considering one class against a group of all other classes.

During the validation process, the classifier chooses a class with a maximum score. And because the data is 100% text, which is often linearly separable (Joachims, 1998) and has many features, the SVM was trained with a linear kernel which in turn makes the classification task faster. The implementation was done in Python and the popular library for machine learning algorithms known as the Scikit-Learn. The LibSVM (SVC) implementation of the algorithm was used, keeping the default values for the regularisation parameter of the error term, C, the kernel and the Gamma function γ.

5.5.2 Random Forest

This is a bagging algorithm. A bagging algorithm is an *ensemble* method that combines prediction from multiple machine learning algorithms together to make more accurate prediction. As a bagging algorithm, Random Forest selects a sample of the instances and a sample of the features from which it then builds the number of individual trees. The sampling is done with replacement. The trees operate as an ensemble. Hence, the Random Forest is considered an ensemble classifier. The Random Forest tries to improve the bagged decision trees. The bagged decision trees select the splitting point using a greedy algorithm that minimises error. The Random Forest algorithm changes this procedure so that the learning algorithm is limited to a random sample of features of which to search. The number of features to be searched was specified at split point, as a parameter to the algorithm, using the following formula.

$$m = \text{sqrt}(p) \qquad (11)$$

Where m is the number of randomly selected features that can be searched at a split point and p is the number of input variables. Each tree in the forest gives its prediction to the sample. The class with the highest votes becomes the model's prediction for that instance (data sample). The fig. 5.3 below shows the class predictions for each tree in the forest.

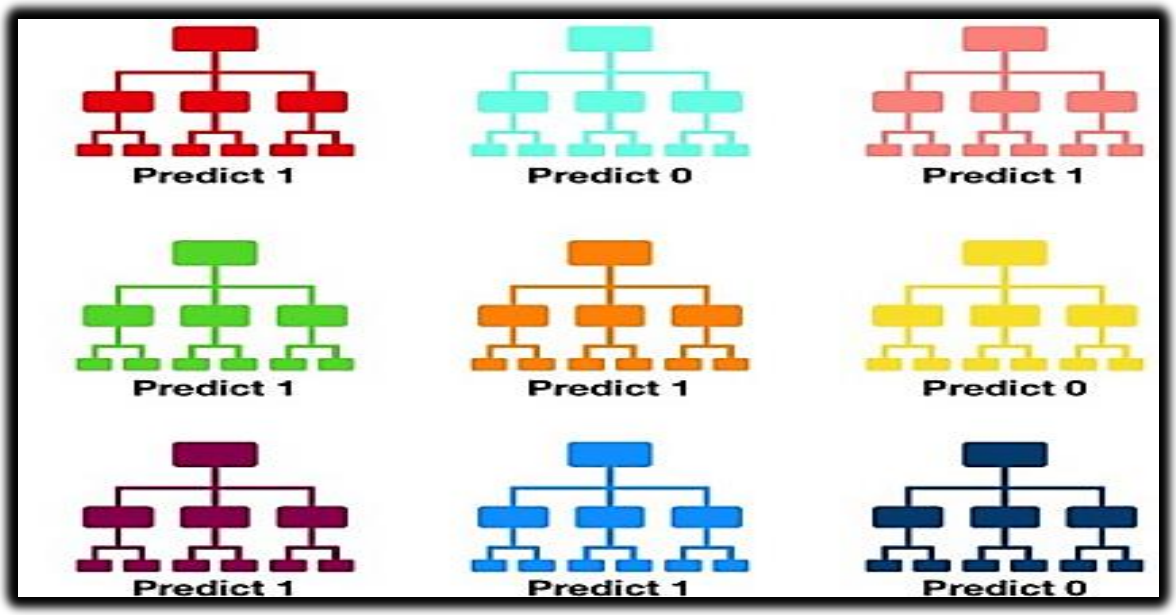

Fig. 5.3 the voting for the classes.

It shows nine (9) trees each producing its own prediction. Six (6) trees gives a prediction of 1 (a positive prediction) while three (3) trees produced a prediction of 0 (a negative prediction). Hence, a prediction of 1 (a majority class) is produced for that instance.

We selected this algorithm for this task for a few reasons:

a. It produces a linear decision boundary that separates the classes.
b. The Random Forest does not over fit, regardless of the number of trees used (except for noisy datasets). This is because the trees were constructed by considering a random subset of the features. This way, the correlation between individual classifiers is also greatly reduced.

The implementation of the algorithm used the default values for the *n_estimators* (number of trees in the forest which is 10) and min_samples_split (minimum number of working set size at node required to split). During each bootstrap sampling, about one-third of the instances are left out of the sample during the construction of the kth tree. This is used for the validation. At the validation phase, we also measured the out-of-bag (OOB) error rates for first n trees. This is the proportion of times that the predicted class (produced by the trees) is not equal to the true class of n averaged over all cases.

5.5.3 Logistic Regression

Just like the previous algorithms, the logistic regression is also a probabilistic and supervised algorithm. It estimates the logistic function given by

$$Logit(P) = \ln\left(\frac{p}{1-p}\right) \quad (12)$$

It represents the natural log of the likelihood (probability P) that a label (category, Y) is assigned to the text or an instance in the data. From the above equation, we calculate the probability P as follows:

$$P = \frac{e^{logit(P)}}{1+ e^{logit(P)}} \quad (13)$$

Finally, the classification of the text into a given class is done by calculating the optimised likelihood of the instance belonging to any of the classes. This is the log-likelihood function. The log-likelihood function is given by.

$$L(X|P) = \sum_{i=1,yi=1}^{N} \log P(xi) \ + \sum_{i=0,yi=0}^{N} \log(1 - P(\text{xi})) \quad (14)$$

All the above algorithms produce a linear boundary classification. They were chosen because the number of classes in our experiment is fixed and relatively small. So, our linear

classification would be computationally inexpensive compared to when the number of classes is big. The computational complexity is ***O(kh)*** where *k* is the number of classes and *h* is the feature dimension of the dataset. The full review of the literature on the above algorithms is reported in chapter two (2) of this thesis. In addition, all of the above algorithms are supervised. This means that they require training and are then tested based on the training.

5.6 Implementation

For the implementation, the Python programming language was used. It has robust support for natural language processing (NLP) libraries. The natural language toolkit (NLTK) is one of the best, robust and most used NLP libraries available in Python. NLTK is effective in text pre-processing tasks such as tokenization, stemming, tagging, parsing etc. Also, the language (Python) has libraries for machine learning tasks. The Scikit-learn library is one of the most effective and vibrant libraries for machine learning. It supports all machine learning algorithms including deep learning algorithms.

NLTK and Scikit-learn libraries in Python were used for the implementations. The classification task was implemented with all the algorithms identified in section 5.4 above, and the results are reported in section 5.6 below.
The data were read and pre-processed. Feature sets were selected from the pool of the text. Each training instance was labelled correctly and accordingly. The feature transformation was performed from the bag-of-words involving the three (3) feature types described previously. These feature sets, along with the labels for each class, were passed to the machine learning algorithms. The data was divided into training set and testing set.

Several training sets were used with testing set ratios. The best results were obtained with the 80:20 ratio for all the algorithms. The 80:20 ratio means that more training was performed and then less testing. Details about the data is in section 3.8 above.

5.7 The Model Evaluation

The models produced by the respective ML algorithms were evaluated to determine their efficiency and robustness for the intended task. Standard machine learning evaluation metrics were employed to evaluate the model. There are several ML evaluation metrics produced for the evaluation of the models. Each metric has its associated strength and weakness. The choice of each metric also has an associated trade-off. These metrics include: Accuracy, error rate,

sensitivity, specificity, precision, recall, F-Measure, Geometric mean, averaged Accuracy, averaged error rate, averaged precision, averaged recall, averaged F-Measure, averaged Geometric mean, mean square error, area under curve (AUC) Etc. The work of Hossin and Sulaiman (2015) has discussed the respective evaluation metrics as well as their strengths, weaknesses and trade-offs. All the existing evaluation metrics are built around how well the model correctly identifies positive or negative classes.

In this research, however, we adopted a novel metric to model evaluation. This is a hybrid approach which combines two or more metrics. By combining the metrics, we take advantage of each metric's strength. There is no fixed rule as to the metrics combination (Ranawana and Palade, 2006). Some situations require a high value of one metric and low value of another or vice-versa, depending on the domain. In our situation, we selected accuracy, precision, recall and F-Measure. The accuracy aims to get a high number of correctly classified instances predicted by the trained classifier. Recall measures the number of positive instances that are correctly classified by the trained classifier. Precision measures the proportion of positive instances predicted from the total instances in the positive class. The idea here is that we want a model that has high precision and high recall. F-Measure represents a harmonic mean between a precision and recall.

The result reported here was arrived at after several experimental runs involving several parameter tunings, and the best run was reported in this experiment. The results of the experimental task are reported below. First, the accuracy was evaluated followed by recall and precision. The F-Measure was also reported. For the three (3) feature types used, the best score (score is given by the feature) is the value reported for each of the machine learning algorithms.

5.7.1 Accuracy

The accuracy of a model is the fraction of the predictions which the algorithms got right. For binary classification, the accuracy is calculated in terms of positive and negative predictions as depicted in the equation below.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (15)$$

• TP = number of the correctly identified positive instance (True Positive)

• FP = number of the wrongly identified positive instances (False Positive)

• FN = number of the wrongly identified negative instances (False Negative)

• TN = number of the correctly identified negative instances (True Negative)

For multi-label classification, however, the technique of one-vs-all (OVA) was used where one class is determined by putting the class on one side and the rest of the classes on the other side. After the training, the model was validated with the validation set. Table 5.6 below shows the accuracies for the four (4) machine learning algorithms over the three (3) features types.

Table 5.6 Accuracies of the algorithms on different feature types

Classifier	Word count (%)	TF-IDF (%)	NGRAM (%)
SVM	71.5	67	52
Random Forest	58.5	64.0	63
Logistic Regression	78.5	73.5	55.5

The scores for all the algorithms over the respective feature types are very close. This is because the six (6) classes have very similar feature (words) values in common. In the experiment, the features are words. Therefore, very many similar words are in common in all the classes, hence it considers the frequencies of the words relative to the size of the vocabulary. The NGRAM feature records the least accuracy value.

The SVM has the least accuracy score over all feature types, with *word count feature* having the highest and NGRAM the lowest. Overall, the Logistic Regression has the highest accuracy score of over 78% (for a word count feature), followed by the Random Forest with an accuracy of 64% (for TF-IDF feature). The SVM and Naïve Bayes algorithms were the least with accuracies of 55% and 61% for *word count* and *TF-IDF* respectively. Overall, the results for the three (3) feature types, the *word count* feature gave a better performance over other feature types i.e. higher accuracy score. Of the three (3) feature types, the NGRAM feature records the least accuracy value. This due to the N-gram nature of the features that looks for contiguous sequence of terms from the document. Failure to find such sequence from the training as well as testing documents resulted in low prediction.

The accuracy alone is never enough to evaluate how good the model is, due to the accuracy paradox. For example, a model that detects all the negative examples but could not detect any or fewer positive examples would still have high accuracy. To evaluate the true effectiveness of our model, we employed other measures (metrics). These included precision and recall.

5.7.2 Precision

This is also known as the positive predictive rate. It measures the fraction of the correctly identified instances. In other words, what proportion of the positive identification was correct? It measures how 'accurate' is the accuracy of the model (reported in section 5.6.1 above). It is calculated as follows:

$$precision = \frac{TP}{TP+FP} \quad (16)$$

The precision was calculated for all the classifiers. We, however, selected the model produced by the best performing machine learning algorithm (the LogisticRregression). The precision recall and F-measure assessment, therefore, only captures the Logistic Regression model. Table 5.7 below shows the precision score for all the classes. For the six (6) classes, the precision of the model has more than 70% precision across all the classes. Just as with the accuracy above, the word count feature has the best numbers for precision.

Table 5.7 Precision for the Logistic Regression

Classifier	Word Count (%)	TF-IDF (%)	NGRAM (%)
Introduction	84	71	55
Literature review	78	68	61
Methodology	71	67	62
Result	76	81	43
Discussion	73	80	52
Conclusion	84	81	57

As seen in table 5.7 above, the best classification precision was with the *word count* feature for all the classes. The *introduction* and *conclusion* classes have the highest precisions of 84% each. This means that the accuracy reported about them is 84% accurate (precise). This is followed by the Literature review with a precision of 78%. Other classes have a precision of 71%, 76%, 77% and 76% respectively. For precision, however, the *word count* feature gives the best precision performance for all the classes.

5.7.3 Recall

This metric measures the fraction of the correct samples picked up by the classifier. Just as with the precision, the recall values were higher with the *word count* feature. The highest precision was with the *conclusion* class with a recall value of 83%. The *resulting* class has

close to 80% recall rate. Just like with the precision, the best recall score is with the *word count* feature for all the classes. Again, this is reflected by the accuracy result reported above. Table 5.8 below shows the recall values for all the algorithms on all feature types.

Table 5.8 Recall for the Logistic Regression

Classes	Word count (%)	TF-IDF (%)	NGRAM (%)
Introduction	62	60	42
Literature review	61	56	57
Methodology	74	67	65
Result	78	66	38
Discussion	75	70	55
Conclusion	83	76	54

The sensitivity or recall measures how good a model is at identifying the positives, while specificity measures how good the model is at avoiding the false positives (negatives mistaken for positives). Precision measures how precise the model is in identification i.e. of all those labelled positive, how many are positive? F-measure combines this parameter together and gives the strength of the algorithm.

5.7.4 F-Measure

This is also known as the F-score or F1-score. It considers both precision and recall. F-score, therefore, is the harmonic mean (average) of the precision and recall. It is calculated as the

$$F - measure = \frac{2*(Precision*Recall)}{Precision+Recall} \qquad (17)$$

The F-measure scores for the all the classes are captured in the table 5.9 below.

Table 5.9 F-Measure

Classes	Word count (%)	TF-IDF (%)	NGRAM (%)
Introduction	80	69	49
Literature review	74	67	61
Methodology	75	74	68
Result	81	79	48
Discussion	78	78	49
Conclusion	83	79	53

The classifier records the highest F-measure performance for the *Introduction, Result and Conclusion* classes with over 80% score. Also, the *word count* feature gives the best result for the F-score measure. This shows that the classifier model is very effective. Other classes of the

canonical model also generated a good performance of over 70%. Overall, the model has a very high precision and recall values, thus, making it a very effective machine learning model.

5.8 Chapter Summary

This chapter presents the result of the machine learning experiment which provides the needed artificial intelligence required to enable machines to identify the various core sections of a research publication. Various machine learning algorithms were tried, including the Support Vector Machines, Random Forest and Logistic Regression. Also, various features types such as *word count, TF-IDF* and *NGRAM features* were used in the experiment. They consider the various feature dimensions of the data. Overall, the Logistic Regression with the word count feature achieved the best result. Therefore, the model produced by the Logistic Regression with the *word count* feature was selected as the model for this project. The models have good accuracies, high precision and high recall. Using the model, each paper can be mapped to the canonical model provided in this paper.

The ML learning methods build models for prediction purposes. The ML methods use varying approaches to learn from the dataset, hence, produce varying degree of performance (results), although each tries to produce prediction as accurate as possible. This is because they use different approaches to learn from the dataset. They also have different assumptions. For example, while SVM considers the vectors as the basis for discriminating between the classes, Random Forest uses votes to decide the classification. So, the varying result is due to the different approach the methods use in the task. The results from all the methods however, satisfactory.

Although, the machine learning models have high precision and recall rates, the numbers (performance score) show that there is room for a better result. Hence, the experiment with deep learning neural networks which is described in the next chapter.

6 The Convolutional Neural Network

In the previous chapter, traditional machine learning algorithms were used to realise the canonical model developed and reported in chapter four (4). The result from the machine learning models is quite good, with a good accuracy and precision of over 70%. With the exponential growth of complex dataset, however, more enhancement in the machine learning methods are required to provide an improvement in the data classification task, providing more accuracy and precision to our classification. Deep learning approaches achieved better results than the machine learning methods in tasks such as image classification, NLP etc. In this chapter, therefore, the same problem was implemented using a convolutional neural network (CNN), which is more sophisticated (deeper). The success of the deep learning methods relies on its sophistication to model complex and non-linear relationship in the data.

6.1 Introduction

Neural networks are algorithms that mimic how the human brain functions. The convolutional neural network (CNN) has been a success in natural language processing (NLP) tasks. However, it has mostly been experimented with small data instances such as a *sentence or question or relation*. The model's performance has not been tested with a bigger data instances such as the size of data instances used for this project. This work, therefore, explored the scalability of the CNN on a bigger data instances such as the scientific research reports, which consists of large text blocks of up to 2,000 words per data instance.

This experiment would establish the robustness and the scalability of CNN with a bigger data input instance.

6.2 Model

The deep learning models map the words in a text to vectors. These vectors are then mapped into a fixed length representation. Fig. 6.1 shows the architecture of the CNN model.

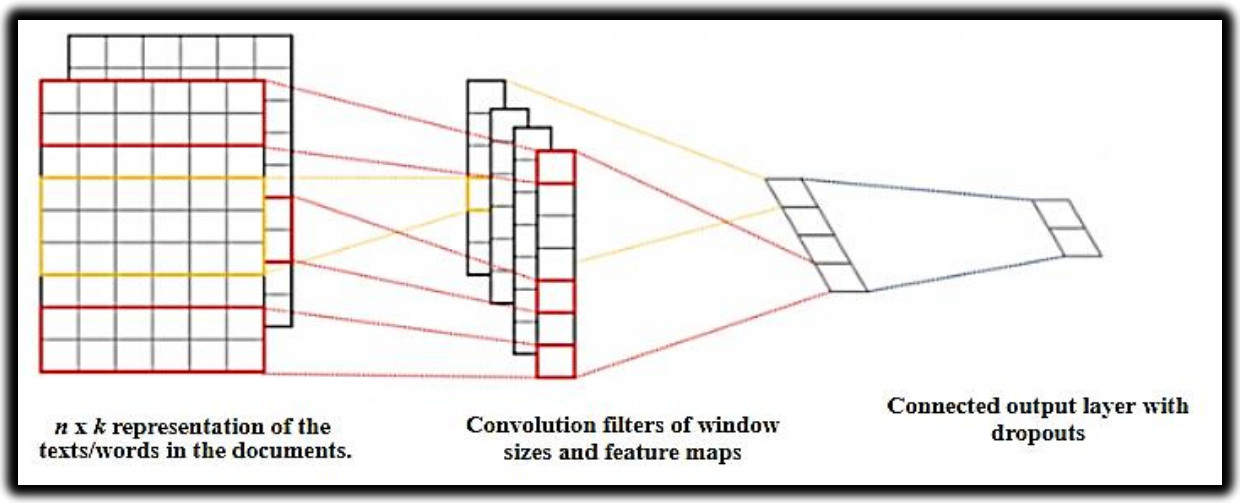

Fig. 6.1 architecture of the CNN model

The CNN consists of the input layer (involving a concatenated *glove* word embedding), the convolutional layers, the max-pooling layer and the output. The data, discussed in section 6.4 below, contains larger input (text) consisting of a few sections which are made up of several paragraphs. The total features of the entire experiment total more than two and a half million

(2,500,000) features. Each section consists of a few sentences. A section of length *n* is represented by

$$y = x1 \oplus x2 \oplus \ldots \oplus xn, \quad (18)$$

Where $\oplus$ is the concatenation function for the individual sentences *x1, x2….xn* that make up the section. Each convolution layer involves the filter operation which involves a filter $w \in R^{hk}$, which is applied to a window of *k* features to produce a feature map. For example, a feature Gi is generated using:

$$Gi = f(w.\sum_{i+k-1} x + b) \quad (19)$$

Here $b \in R$ is a bias term and *f* is a non-linear function such as the hyperbolic tangent. The expression $\sum_{i+k-1} x$ is the summation of *k* words in every section (i.e. the number of words in all the sentences in the section). This filter is applied to each possible window of words in the sentence $\{x_{1k}, x_{2:k+1}, \ldots, x_{n-k+1:n}\}$ to produce a feature map in each layer. Every layer in the network acts as a filter (to check) for the presence of specific features or patterns present in the original data using the formula in equation (21) above. The first layer in the network detects many features. Increasingly, subsequent layers detect smaller features, which are more abstract and are usually present in the features detected by the other earlier layers. The final layer in the network can make the classification/prediction by combining all the specific features detected by the respective layers (filters) in the data.

The max-pooling operation is then applied over the feature map and takes the maximum value $\hat{G} = \max\{g\}$ as the feature corresponding to this filter (Kim, 2014). This is done such that the highest value filter (the most important feature) is captured for each feature map. By default, this pooling scheme naturally deals with variable lengths of the respective documents. The model uses multiple filters (with varying window sizes) to obtain multiple features used for the experiment.

$$g = [g_1, g_2 \ldots g_{n-h+1}] \quad (20)$$

Where $g \in R^{n-k+1}$.

6.2.1 Feedback loop

The networks are entirely feedforward. However, as the model learns from the data, errors are generated. Feedback loop is for propagating the error forward through the network. However, with Error Forward-Propagation the network only has feedback loops, from the output neurons to the input receiving neurons, which are treated as feedforward connections. The network makes predictions in only one direction, forward. The learning (training) is an iterative process. This is because a limited dataset is used to optimise learning. Each cycle is called is called an epoch. For every epoch (one complete cycle), the network performs both feed forward and back loop. The forward learns from the data and the backward updates the weight for the next forward loop. Thus, the iterations continue until the learning does not improve.

6.3 Regularisation

One basic problem in machine learning, including deep learning, is the overfitting problem. Overfitting refers to the model that trains on the data too well, such that it affects the prediction on the new (unseen) data (Bengio, Courville and Goodfellow, 2016). In deep learning, regularisation is a common and efficient method to avoid overfitting. There are a number of regularisation techniques in deep learning. Dropout is one such efficient regularisation technique. At every iteration it selects and removes some nodes in the network along with their incoming and outgoing connections to such nodes. The fig. 6.2 below shows the dropout network.

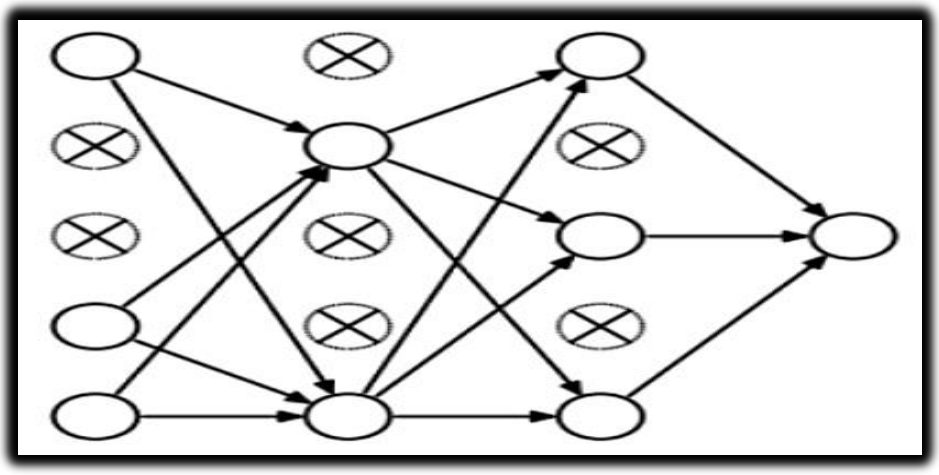

Fig. 6.2 dropout from the network.

So, each iteration results in a different set of nodes. This is like an ensemble technique in machine learning. For our experiment, we used the dropout technique for regularisation. This

is because it randomly drops out the nodes, i.e. sets the nodes to zero during the forward propagation. For output y in a feedforward propagation, dropout:

$$y = w \cdot (z \circ r) + b \tag{21}$$

Where ◦ is the element-wise multiplication operator and r ∈ R is a 'masking' vector of Bernoulli random variables with probability p of being up to 1. Gradients are back propagated only through the unmasked units. At test time, the learned weight vectors are scaled by p such that $w\hat{} = pw$, and w^ is used (without dropout) to score (predict) unseen documents.

The weight of the network was larger than normal because of the drop out. Therefore, the weights were scaled by choosing the dropout rate. The dropout rate is a floating fraction between 0 and 1 on the inputs at each update.

6.4 Datasets

The same dataset used for the experiment reported in chapter five (5) is used for this experiment. The training data consist of six (6) classes each with a total of one thousand (1,000) documents, totalling six thousand (6,000) training documents. Each document is a text file containing data which corresponds to the identified class. Table 6.1 below shows the detailed data description.

Table 6.1 Average data documents

S/N	Class	No. of documents	Total Doc size (words)
1.	Introduction	1,000	606,700
2.	Lit. review	1,000	773,345
3.	Method	1,000	1,077,745
4.	Result	1,000	895,640
5.	Conclusion	1,000	229,023
6.	Discussion	1,000	209,615

The training and validation were done as described in the section below. The data amounts to more than eight million parameters (features). As discussed previously, most researches use short-text data to work with CNN in NLP. Our research, however, applied a larger dataset.

Hence, our research also investigated the effectiveness of CNN in NLP tasks with bigger data instances.

6.4.1 Training

From the data, the top 3,515,881 trainable parameters. 80:20 ratio was used for the training and validation. We also used a filter window (*k*) of 5 each with a filter size of 128. The maximum sequence length of 1,000 was also used. Dropout rate of 0.5 and batch size of 2 were also used. The accuracy was used as the metric for measuring the e model. The details of the data are shown in the fig. 6.3 below.

Simplified convolutional neural network

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	(None, 1000)	0
embedding_1 (Embedding)	(None, 1000, 100)	3270500
conv1d_1 (Conv1D)	(None, 996, 128)	64128
max_pooling1d_1 (MaxPooling1	(None, 199, 128)	0
conv1d_2 (Conv1D)	(None, 195, 128)	82048
max_pooling1d_2 (MaxPooling1	(None, 39, 128)	0
conv1d_3 (Conv1D)	(None, 35, 128)	82048
max_pooling1d_3 (MaxPooling1	(None, 1, 128)	0
flatten_1 (Flatten)	(None, 128)	0
dense_1 (Dense)	(None, 128)	16512
dense_2 (Dense)	(None, 5)	645

Total params: 3,515,881
Trainable params: 3,515,881
Non-trainable params: 0

Fig. 6.3 Trainable and non-trainable features

6.4.2 Word embedding

The word embedding is the conversion of the textual data into numbers that would be passed to the CNN. It is a kind of transfer learning for words. Rather than calculating the word embedding, we used pre-computed word embedding. They are a fixed size pre-trained set of vectors. For our training, we used *Globe* embedding, the global vectors for word representation by the Stanford Group. It also ensures a semantic representation of the words. The *Globe* embedding consists up to 400,000-word vectors with the dimensionality of 100 and was trained using the continuous bag-of-words architecture (Mikolov *et al.*, 2013). Words that are not present in the *Globe* embedding are initialised by randomly sampling from a uniform

distribution in [−0.1, 0.1] (Zhou *et al.*, 2016). The word embedding is fine-tuned during training to improve the performance of classification.

6.4.3 Software Packages Used

The software used for the experiment consist of python programming language which has robust libraries for the machine learning tasks, including deep learning. The implementation was also integrated with Jupyter, the open source and open standard for interactive computing in many programming languages including Python. The Pycharm and Anaconda IDE were both used independently. The result was, however, not affected by the different IDEs used.

6.5 Result and Discussion

Table 6.2 below shows the result of the CNN model with respect to the 6 classes. It shows the epochs and the accuracy results of the training and validation. The difference between successive values is due to the improvement as the weights are updated in the feed forward network. Overall, a training accuracy of 92% and validation accuracy of 85% were respectively achieved by the model for each of the classes. 15 epochs were used in the experiment, i.e. the weights of the training and the validation vectors were updated 15 times in a feed-forward manner to ensure the improvement of the model. After the 10^{th} epoch, the validation accuracy of the model remains the same, hence, the final value of the validation accuracy is reported in this research. The table 6.2 and fig. 6.4 below shows the score and visualises the training and validation accuracies respectively.

Table 6.2 Accuracy values

Epoch	accuracy	Validation accuracy
1/15	0.3969	0.4425
2/15	0.5485	0.6235
3/15	0.6625	0.645
4/15	0.7250	0.6566
5/15	0.8250	0.6451
6/15	0.8240	0.6500
7/15	0.8975	0.7440
8/15	0.8625	0.7894
9/15	0.8750	0.7500
10/15	0.9250	0.8000
11/15	0.8875	0.8230
12/15	0.9250	0.8450
13/15	0.9750	0.867
14/15	0.9375	0.8720
15/15	0.9250	0.8451

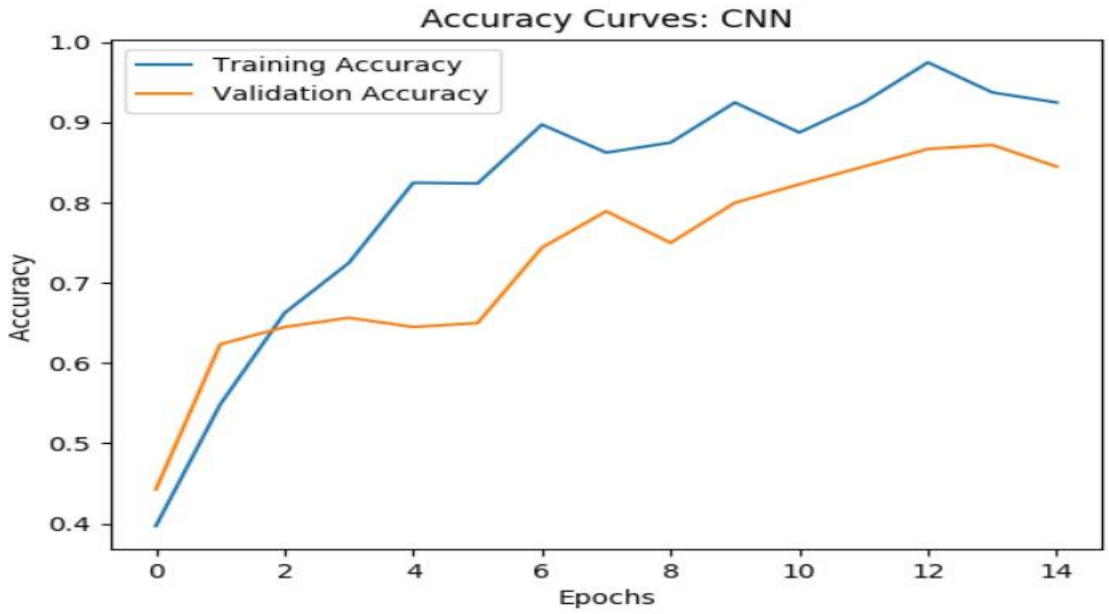

Fig. 6.4 the accuracies curve

6.5.1 Loss function

The loss function shows the error reduction in the training and prediction. This determines the strength of the model. It depicts how our model evolves as the training progresses. The lower the loss values the better the model becomes. Table 6.3 below shows the values for the training loss and validation loss. The table 6.3 and fig. 6.5 shows the score and visualises the loss function respectively.

Table 6.3 the loss values

Epoch	Training Loss	Validation Loss
1/15	1.4680	1.3072
2/15	1.1499	1.3001
3/15	1.1115	1.2919
4/15	0.9451	1.2873
5/15	0.7497	1.1414
6/15	0.7034	1.5273
7/15	0.6452	1.2248
8/15	0.4755	1.0078
9/15	0.4063	1.0055
10/15	0.4011	1.3590
11/15	0.3737	1.2219
12/15	0.2344	1.1734
13/15	0.1823	1.0924
14/15	0.1431	1.1138
15/15	0.1814	1.3792

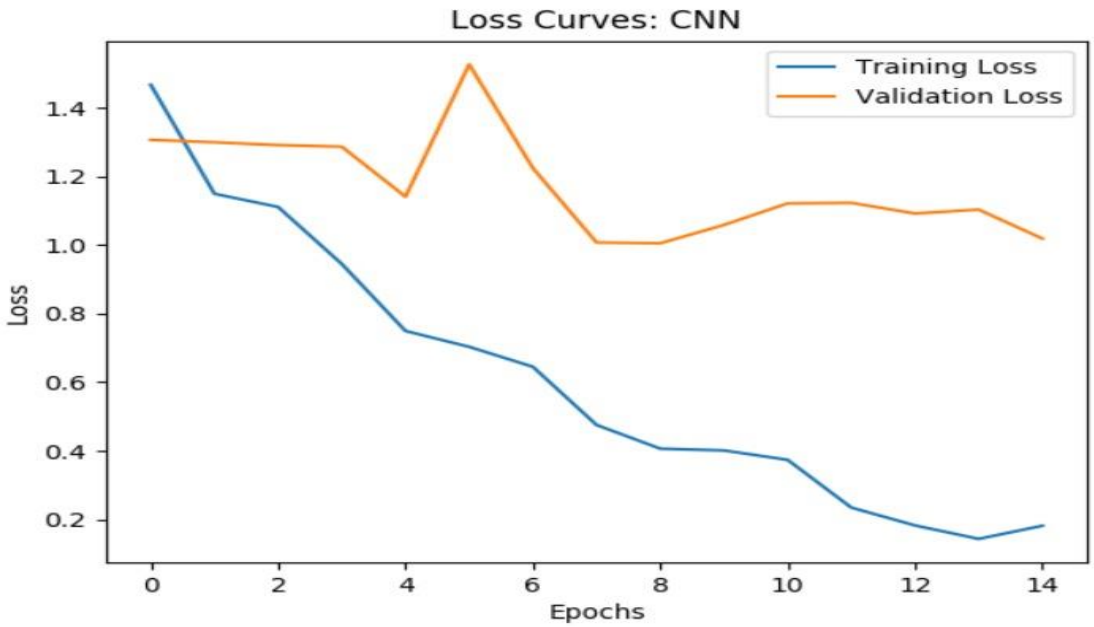

Fig. 6.5 the loss function

As the number of epochs increases, the loss function (the inconsistency between the predicted and actual label) for the training decreases. The validation loss also decreases with the increase in the number of epochs. Fig 6.5 above shows the visualised relationship between the training loss and the validation loss for the model. At the start of the computation, both the training loss and the validation loss were quite high. This is as a result of not enough learning for the model. However, as the weights are updated in the model, the losses begin to fall resulting in a more fitted model. As can be seen in the fig. 6.5 above, the training loss is lower than the validation loss as the number of epochs increases. This is because the validation consists of the data which has not been used in training the model and the model tries to predict the class of the new data during the validation. It has been excellent in most part, but the gap between the orange and the blue line explains the discrepancy between the two (2) scenarios. However, the loss function is at an acceptable level at the 10^{th} epoch after which the validation accuracy stalls. Hence, the model is well fitted for the dataset. In other words, the model performed quite well when later applied to unseen data.

In each round of learning from our dataset, the learning improves with more epoch. The learning, however, is not directly proportional to time or epoch. During the first few epochs, the learning (accuracy) improves, while the loss (error) reduces. After the 10^{th} epoch however, the learning ceases to improve and the loss (error) stops reducing. After this point, the accuracy begins to decline and the error increases. That was the point at which the experiment was stopped and the values (results) were reported.

6.6 Comparison of CNN with other models

Earlier in this project, the same problem was implemented using the traditional machine learning algorithms such as the Naïve Bayes, Support Vector Machines (SVM) and the Logistic Regression, using the same dataset and same training-validation ratio of 80:20.

From the results of the machine learning experiment reported in chapter (5) of this thesis, the CNN results outperformed the traditional models. The application of CNN in NLP is an effective technique especially for the text classification. Table 6.4 shows the results of the various algorithms.

Table 6.4 Results of the various models

S/N	Model	Accuracy
1.	SVM	55%
2.	Naïve Bayes	64%
3.	Logistic Regression	78.5%
4.	CNN	85%

6.7 Chapter Summary

In this work, we implemented a text classification problem involving a data with a large set of input features, 6 classes and using the convolutional neural network (CNN) built on top of *Globe* word embedding. The CNN model achieved an accuracy of 85%. This work is the first to try the CNN NLP task involving a large and complex data set such as the full text of research publication (journal/conference proceedings). The result also shows that CNN performs better than the traditional machine learning when the data instances are big/large.

7 Summarisation

In chapters five (5) and six (6), the machine learning and deep learning (convolutional neural network) models were used to produce the machine intelligence needed to enable the actualisation of the canonical model (the proposed approach for automatic data extraction produced and reported in chapter four). A goal of the research requires that the desired section from a document would be located and presented to the reviewer in a concise (summarised) form. This chapter presents the development and implementation of the summarisation method. An automatic tool (ROGUE) was used for the automatic evaluation of the system generated summaries alongside the human evaluation of the summaries.

7.1 Summarisation Approach Used

As detailed in chapter two (2), the process of summarisation has two (2) approaches: extractive and abstractive techniques. When a reviewer reads a paper, he/she would like to see the original text as reported in the documents under review. For this reason, the extractive method was selected for use in this research. The fig. 7.1 below shows the extractive summarisation process used in this research.

7.2 Summarisation Process

The process involves converting the text to sentences, pre-processing it, comparing the sentences, scoring the various sentences for relevance to the summary, sorting and selecting the sentences and finally putting the final summary as the representative of the original document(s). Various extractive summarisation approaches were tried. However, the process of implementation remains the same for the approaches used in this research. The original data can be a single or multiple document. The process is depicted in fig. 7.1

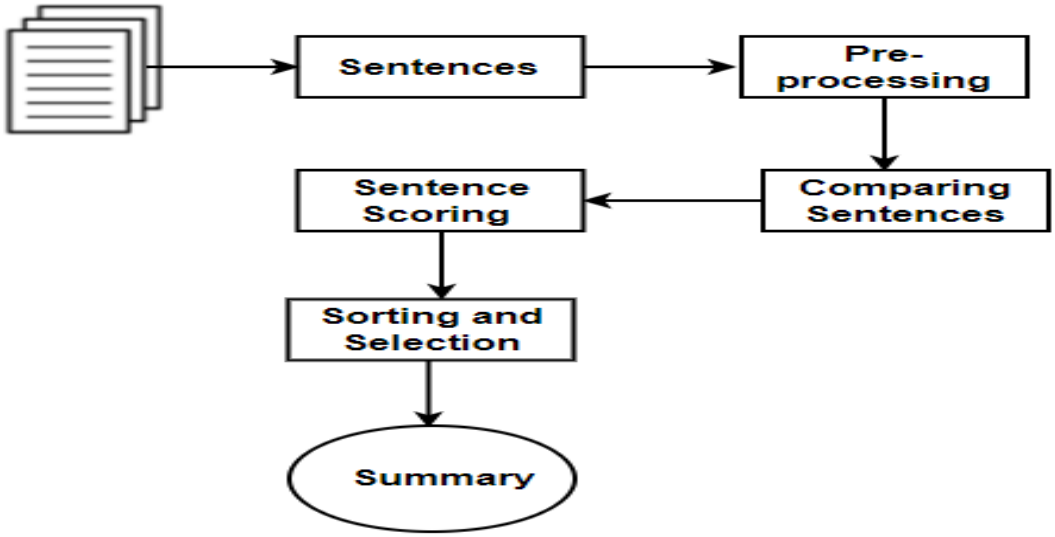

Fig. 7.1 the summarisation process.

7.2.1 Tokenization

After pre-processing the text to clean noise, the texts were tokenised into sentences and then into constitute words. The words are then scored for each sentence. The score of the constituent words were then summed to give the score for each sentence. A sample of the tokenization of the text into sentences is shown in table 7.1 below. The sentences are then converted into individual words for scoring.

Table 7.1 sentence tokenization

Text	Sentence tokens
However, to score these sentences, we break the sentences into constitute words. The words are then scored. For each sentence therefore, we score of the constituent words are summed to give the score the score for that sentence. Before scoring, the words (sentences) we did some pre-processing to clean the text from noisy and irrelevant tokens.	❖ However, to score these sentences, we break the sentences into constitute words. ❖ The words are then scored. For each sentence therefore, we score of the constituent words are summed to give the score the score for that sentence. ❖ Before scoring, the words (sentences) we did some pre-processing to clean the text from noisy and irrelevant tokens.

7.2.2 Pre-processing

This is the first stage in any data analytics. The pre-processing includes the removal of special character, numbers and a single lettered word. It also includes the removal of white spaces and stop words. From the implementation perspective, the pre-processing stage converts the text from the natural (human) language to machine understandable format for further processing. The pre-processing is important for a cleaner and efficient result as irrelevant bits (stop words) may affect the summary. Stop words are the commonly occurring words like “a”, “in”, “on”, “is”, “was” etc. These words do not carry important meaning and are usually removed from texts. In our implementation, we used the Python’s natural language took kit (NLTK) library for the tokenization task.

7.2.3 Comparison of the Sentence

In this stage, each sentence was mapped against the bag of words, constructing the frequency matrix of each word in the sentence. This is implemented as key: value for each sentence. Each sentence is the key while the value is a dictionary of word frequency.

7.2.4 Sentence Scoring

In this stage, each word in the sentence is scored in relative importance to the document under consideration. The score for each sentence is then computed as a function of the words in that sentence. The scoring is done using the various approaches discussed in the literature review in chapter two (2). Section 7.4 gives the details of the various scoring methods and their implementations.

7.2.5 Selection and sorting of Sentences

After calculating the score for each sentence, the sentences are then sorted according to a value (total score) of their relevance to the document. The sorting is done in descending order. The candidate-sentences are then selected based on these scores. The number of sentences required to generate (build up) the summary is specified by the user. The top-ranked number of sentences, corresponding to the user-specified number, are then collected as the summary candidate sentences. For a meaningful summary, convenience and flow in the story, the sentences are then re-ordered according to their position in the original document(s) (Zhao, Jiang and Liu, 2019). These re-ordered sentences are the required summary, produced by this method (process).

7.3 Implementation

The process depicted in section 7.2 above is implemented using the various summary methods highlighted in chapter two (2). The implementation was done in python programming language. Python has library support for the various summary methods and the natural language processing toolkits in general. After the implementation, the method with the best summary was chosen as the summariser for this project. The evaluation of the best method was done by the automated tool, the ROGUE and the various potential users of the system.

7.4 Result of the various summary methods

The summarisation methods are basically of two (2) groups: the single document and multi-document approaches. Single document approach takes a single document as input and produced the summary while the multi-document approach takes a number of documents and produces a single summary for the collection. This research involves multiple documents for which a single summary was generated. Both approaches were applied and evaluated. For

single document approaches, the summary of each document is produced and then a summary of the various summaries is produced. The result of each approach is given below. The documents used for the summary are attached in the appendix A-F. The document is about one page in length.

7.4.1 Frequency Based Approach

In this approach, relevant sentences are identified by frequency of relevant words. The relevant words are identified using the word probability or the frequency of occurrence of such words. This is calculated as the Term Frequency-Inverse document frequency (TF-IDF) given by equation 11 in Chapter 5. The TF-IDF for each word in the document is computed for all the terms in the document. A sample of the terms along with their TF-IDF scores is shown in appendix F(A).

Using these scores, the sentences were ranked. For each sentence, the TF-IDF for each individual word is summed up as a relevance score for that sentence. The sentence score is computed using the equation 23 below.

$$\sum_{i=1}^{n} wi = w1 + w2 + w3 + \cdots + wn - 1 + wn \qquad (22)$$

Where wi is the word at position ith in the sentences.

For every sentence in the documents, its score is computed using the equation 25 above. The sentences are then sorted according to their scores. From the sorted grouping of the sentences, the top-k sentences, where k is determined by the user. Hence, the user determines the most important sentences as the candidates for the summary. The sentences were finally re-ordered to reflect the order in which they appear in the original document(s). From the computations, the top 20 sentences were extracted and re-ordered to generate the summary shown appendix F(B).

7.4.2 Graph Based Approach

This approach considers the document as a graph, where the sentences are the vertices while the edges are the similarities (weight) of the sentences. The LexRank and TextRank are the two (2) implementations of this approach. We implemented both using Python programming

language. Python has an in-built implementation of these algorithms in '*sumy'* package with Plaintext Parser, NLP tokenizers and summarisers libraries (modules).

The overall idea of LexRank algorithm for summarisation is diagrammatically represented in the fig. 7.2 below.

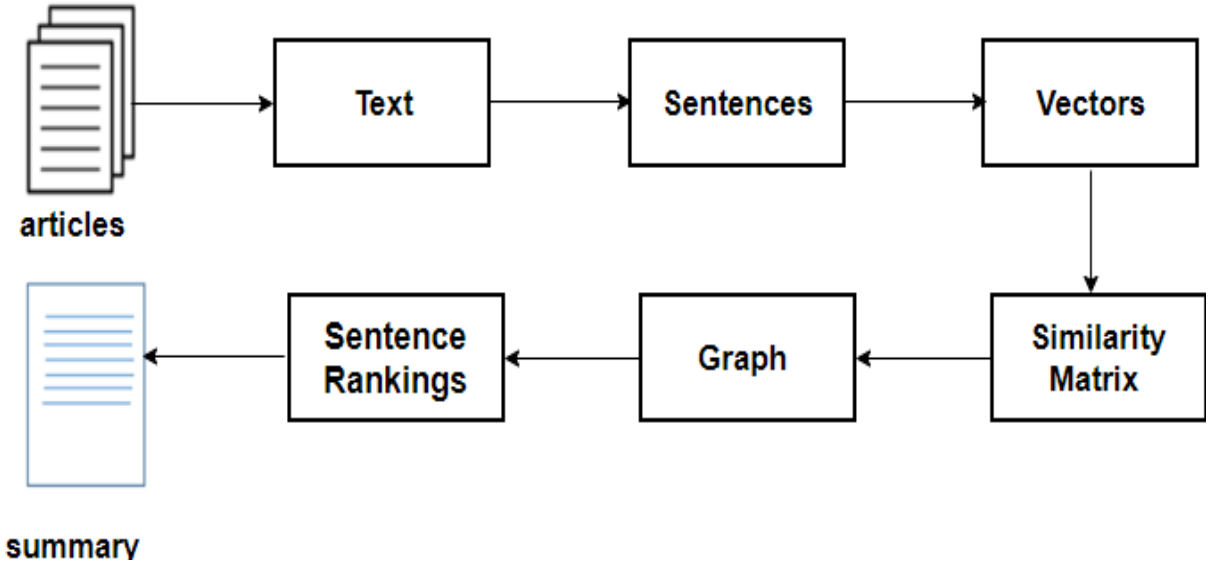

Fig. 7.2 Graph based approaches process

First, the text from the document is pooled together as raw text and then tokenized into sentences. The vector representation, i.e. word embedding, is calculated for each sentence in the collection by calculating the similarities between the word vectors. The similarity between the sentences are measured using the *cosine similarity* measure. The similarity matrix is then converted into a graph, with sentences as vertices and similarity scores as edges, for sentence ranking. Each sentence is then connected with other sentences. Every sentence has a corresponding ranking score. The ranking is in descending order with higher similarity score at the top. The sentences with more connections are ranked above those with fewer connections. Finally, we extracted the top ten (10) ranked number of sentences from the ranking pool to form the final summary. The implementation yields the result (summary) shown in appendix F(C).

The TextRank implementation is based on the PageRank algorithm for web pages. In our context of text summarisation, sentences are ranked based on the number of other sentences that link to them via common words, thus indicating their level of relevance. The linkage between the sentences is depicted in fig. 7.4 below.

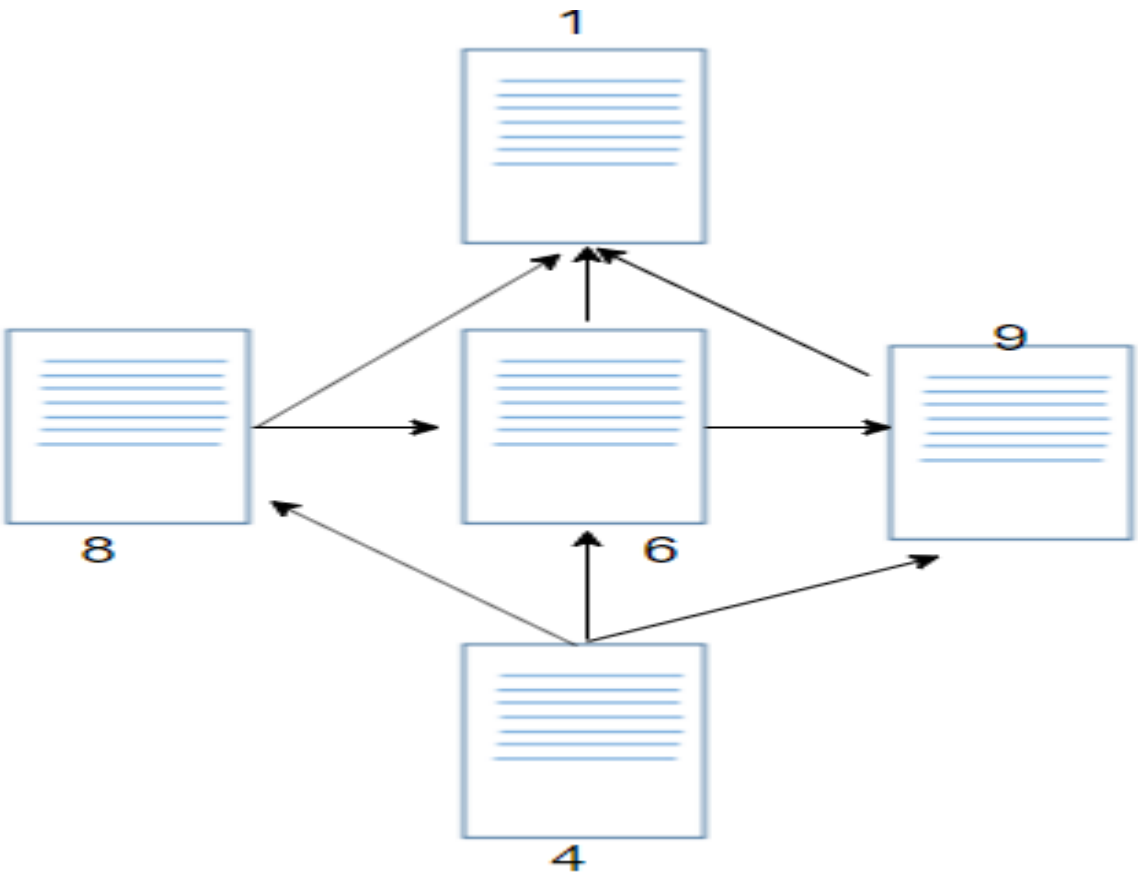

Fig. 7.3 Page Ranking scheme.

PageRank algorithm for webpages is the same concept in TextRank, where text is used in place of webpage. In PageRank, web pages are linked together by common words. In the context of summarisation, however, TextRank links sentences together by common words. Each arrow points to another sentence with at least a common word. The more the connections between the sentences, the more relevant they are. TextRank implementation yielded the summary (summary) shown in appendix F(D).

7.4.3 Cluster Based Approach

This technique is just like the previous ones. The difference, however, is that this approach is based on the concepts of clusters. It generated a centroid of clusters for common sentences. Similar sentences are grouped together in same clusters. The similarity between sentences is computed using the *cluster centroid* method. From each cluster, the top-ranking sentences are selected to form the summary. The number of sentences extracted are also determined by the users. This approach works for both single and multi-document. For this implementation, however, we used single document approach. This means each document is summarised independently. The implementation of this method resulted in the summary in appendix F (E). Like the other approaches, we fetched the top 20 sentences to form the summary.

7.5 Evaluation Using ROUGE

The Recall-Oriented Understudy for Gisting Evaluation (ROUGE) is a tool for evaluating the automatic (system-generated) summaries of texts. It is also used for evaluating t machine translation. It works by comparing the automatically generated summary to the reference summary. The reference summary is manually generated by the users. In this case, a reference summary was carefully generated. Since the extractive approach is being used, the reference summary is generated by identifying the relevant sentences. The same length of summary was generated manually as the automatic summaries (10 sentences). Appendix F (F) shows the reference summary.

The system generated summary was evaluated using two (2) evaluation metric: Precision and Recall. The precision and recall are the two (2) metrics used to measure the accuracy of the automatic summarisation, both relying on the underlying words between the reference summary and the system generated summary.

7.5.1 **Precision and Recall in ROGUE Context**

The recall measures how much of the reference summary is captured in the system generated summary. It is calculated by comparing the number of overlapping words between the systems generated summary and the reference summary. It is calculated as follows

$$\text{Recall} = \frac{number\ of\ overlapping\ words}{Total\ words\ in\ reference\ summary} \qquad (23)$$

On the other hand, the precision measures how much of the system summary is relevant or needed. It is calculated as follows:

$$\text{Precision} = \frac{number\ of\ overlapping\ words}{Total\ words\ in\ system\ summary} \qquad (24)$$

The various system generated summaries were scored (by recall and precision) as follows. A small python script was developed to perform the identification of the overlapping words.

Table 7.2 the Evaluation scores

Summariser	overlapping words	System summary words	Reference summary words	Precision	Recall
LexRank Approach	175	276	272	0.63	0.64
TextRank Approach	**219**	**372**	**272**	**0.65**	**0.81**
Frequency Based Approach	54	106	272	0.51	0.20
Cluster Based Approach	161	256	272	0.63	0.61

7.6 Discussion

From the respective summaries produced by the summarisation approaches, an evaluation was carried out using the ROUGE tool recall and precision rates. As table 7.2 shows, the TextRank approach has the highest recall rate of 81%, followed by LexRank approach. The frequency-based approach has lowest recall rate of 20%. The TextRank has a better recall rate because it measures the relevance of the text before generating the summary. The feature-based approach recorded the lowest recall rate. It only measures the frequency of occurrence of the words without consideration of their relevance with respect to the entire document. The precision of the TextRank approach is also the highest. This means that the 81% recall is 65% precise. The reference summary was generated by the researcher. That summary may not be the same for another group of researchers. For this reason, the system is evaluated also by other researchers.

TextRank was chosen as the summariser to complete the task of the data extraction in the SLR framework.

7.7 Chapter Summary

Four (4) summarisation approaches: TextRank, LexRank, Cluster-based and Frequency-based were evaluated for this research. Various summaries were produced by the approaches with varying degree of accuracies. The ROUGE tool was used to evaluate the various summaries. The TextRank approach produced the best summary with best recall and precision. This is, however, a machine-based evaluation. A more extensive evaluation was carried out with potential users and is described in the next chapter.

8 Human User Evaluation and Verification

In this chapter, we report on the outcome of the hybrid approach. In the hybrid approach, human actors participate in the verification of the machine learning model verification as a hybrid approach to AI-NLP research. The entire project is evaluated. This includes the individual (main) components of the project (machine leaning and the summarisation). The human actors verified the outcome of the computational and automated models. This involves feedback through evaluation forms and interviews on the overall system pointing out the areas of strength and weakness.

The exact task as well as the instructions on how to accomplish such task is detailed in subsection 8.3 below.

8.1 The Hybrid Approach

Most scientists use computational and automated models with excessive experiments and simulations to run analysis as well as evaluate the performance of the methods without the involvement of human actors for verification. In this research however, humans as subject experts were involved. This is to help label the data in order to train the model and to help validate the results produced by the machine learning models. In this way, the feedback was taken to label the data, test the results and further optimise the model.

The user evaluation took two forms:

1. Participants tested the system with their own selected documents and provided responses of through a form
2. Interviews were conducted with three subject experts

We recruited subject experts who have background knowledge and experience in review process, information extraction as well as the machine learning to take part in the evaluation. The task of the participants is to give a professional feedback on the models developed and the computational (experimental) procedure followed. The assessment also identified the strengths and weakness of the methods as well as where they can be improved. Feedback forms were used to capture the responses from the participants. The evaluation by the expert was used optimise the model by optimising the feature sets in the training dataset, parameter tunings etc. The full description of the of the recruitment, evaluation and assessment forms is attached in the appendix A, B, C, D and E.

8.2 The Interview

For the interview, three domain experts were interviewed. All of them have expertise and years of experience in the subject. The first expert interviewed is from the National Information Technology Development Agency (NITDA Nigeria). The second is in the Faculty of Engineering, Environment and Computing, Coventry University, United Kingdom. The third is from the School of Computer Science and Digital technologies, University of East London and the co-founder of a software development company. All the experts have PhD in the subject. Two (2) have more than ten (10) years of post-PhD experience in the field while the third has more five (5) years of post-PhD experience. Due to the COVID-19 pandemic, the

interview was conducted remotely (online). The following is the format and nature of the interview.

1. Seven (7) experts were identified (through personal and professional connections) and contacted. Only 3 agreed to take part in the interview, each at his convenient time.
2. Same questions were asked to each of the experts. The answers were recorded.
3. The project brief (the details of the research aims, objectives and the outcomes as well the evaluation task expected from them) was shared with them. A demonstration of the various implementations of the machine learning algorithms and the output models was also made.
4. Where necessary, the researcher also provided more information and clarification regarding the project and or the system.

At the end of the demonstration session, the following questions were asked.

1. Suitability of the chosen machine learning algorithms for the given task and the dataset.
2. The approach to the problem. The research considers the task as a classification problem. Is this approach appropriate?
3. The feature engineering is based around three (3) feature types: word count, N-gram and the TF-IDF. Is this the right feature engineering approach?
4. The experimental settings and assumptions (experimental parameters) are mostly taken at the default settings. Where the result was not convincing, parameter tuning was performed to enhance the result. Is this the professional approach? Any comments?
5. Considering the respective results from the various machine learning algorithms, any abnormality in the result?
6. On the overall, are the results from the respective algorithms satisfactory?
7. The metrics applied for the evaluation purposes, are they the right choices?
8. Any other comment on how to improve the system

There is high level of satisfaction with the outputs from the various components of the project and the project. This includes the satisfaction with the choice of the machine learning methods, the feature engineering, the evaluation metrics and the training/validation process. However, all of them have concluded that an improvement is required. Suggested areas for improvement include more training dataset, using higher N-grams such as 2-gram, 3-grams could result in improvement. Similarly, using scientific version of TF-IDF to capture only the terms

(keywords) specific to science could make the training better. The notes (responses) from the interview have been captured in the appendix G.

8.3 Evaluation by participants

The participants were given the instruction in print version. They read the instructions carefully before beginning the evaluation exercise. The following were the tasks, along with the sequence, for the exercise.

1. The system is already setup. No need for any configurations or changes to any part of the system.
2. As they must have been informed in the consent form and participant information sheet which are attached in the appendix, participants must have brought their own piece of document to be used for the exercise. This is the preferred option. However, the primary researcher can offer an alternative arrangement should they require.
3. The system evaluation is in two (2) parts. Each part works on the same piece of document.
4. Save the document on the location accessible by the tool for the evaluation. The researcher would help put the document in the appropriate location accessible by the tool.
5. All the needed pre-processing would be done or assisted by the researcher who would be there throughout the duration of the exercise which would last for approximately 20 minutes.
6. The tool would attempt to pick/identify two (2) sections from the paper. These are the ‘methodology’ and ‘result’ sections. This is the machine learning aspect of the system.
7. Look carefully at the output produced by the tool for accuracy or otherwise.
8. The next part of the system would attempt to summarise this identified section to give the information from that section in a concise form.
9. Rate the summary produced by the system using either the summary you produced or your intuitive judgement.
10. Complete the evaluation form you are provided with. The form is named ***Form EF1***.

If anything is unclear, feel free to ask the primary researcher for clarification regarding any part of the system.

8.4 Participant Recruitment

The participants recruited for this task have experience in research, review of literature, as well as machine learning skills. All of them have conducted the review exercise at least once. Staff and students of Coventry University were recruited for this task. The table 8.1 below implies all have PhDs. Since, this project is about systematic review in the software engineering domain, a broad view was taken of what constitutes the software engineering domain including design and innovation. Participants were from the School of Engineering, Environment and Computing, particularly the computer and mathematics department. Because review of literatures across all disciplines share a common goal, participants were also recruited from other disciplines, for instance engineering and social sciences. In total, 30 participants took part in this evaluation, consisting of 25 PhD students and 5 academic staff members of the institution.

The research identified the potential candidates (participants) and then sent them an invitation note. The identification and invitation were implemented via one or all the following methods. The research verbally talks to the potential participants for the consent to participate in the exercise. This was mostly for the candidates already known or accessible to the researcher. An email invite was sent to some identified candidates. The email details the purpose and the specific tasks expected of the candidates. This method is for referrals from colleagues or the network made during the research.

The supervisory team also identified some of the participants. They were invited to the table for the evaluation task. Only participants that replied to the invite were given the paperwork to read and sign for their consent to take part in the exercise. The participant information sheet and the consent forms are attached in the appendix. The table below shown the background information for the participants.

Table 8.1 showing the participants' information

Gender		Experience		Qualification		
Males	Females	< 3	>= 3	PhD	M Sc	Others
30	0	14	16	30	0	0

8.5 The Period of the Exercise

The invite sent to the participants also included the period of time proposed by the researcher for the evaluation task. The proposed period was 3 days from 9th December 2019 to 12th December 2019. Not all the participants, however, agreed to this date. A compromise was made to accommodate various times indicated by the participants. In total, the exercise took a total of 7 consecutive days (5 working days and 2 weekend days) from the 9th December 2019 to 15th December 2019. This period allowed the research to cover as many target users as possible.

8.6 The Evaluation Questions

For the evaluation, some questions were drafted about the task/project. The purpose of the questions is to determine if there is enough statistical evidence in favour of the project. The evaluation will focus on the key individual components and the project. The individual components include the machine learning and the summarisation components to be tested by the users. The machine learning component was tested using question I below. The summarisation component was tested using question II.

Question I: Is the machine learning identification fit for purpose?
Question II: Is the summary generated by the tool enough?

The detail of the evaluation was fully specified in the project instruction (brief). Each participant read the instruction prior to starting the exercise. The researcher also answered questions from the participants. With the help of the researcher (where needed), each participant carried out the various specified tasks and answered questions based on their experiences from the exercise. To evaluate this system, the various responses given by the participants were assessed. Appropriate data analysis tools were used to make the assessment of the various responses. The response form and the instruction for the participants are contained in separate documents.

8.7 Result of Evaluation

The tables 8.2 and 8.3 below show the various responses captured from the participants. In total, 30 participant responses were captured. For each section, we tabled the responses as shown in subsections below.

8.7.1 Machine Learning

The table 8.1 below shows the success of the machine learning models for prediction (on the unseen data). Most users supplied their own data. Others, however, used the data supplied by the project team. The users experimented with only two (2) classes of the data, which are the *result* and *method* classes. And since we have both traditional model and deep learning models, users first used the traditional model. If the traditional model is unsuccessful, the users then tried the deep learning model. If any of the models is successful, then the result (score) is successful. The table 8.1 below shows the result of the users' experimentation. There are 18 successful identification and 12 unsuccessful identifications, each represented by 1. Since there were thirty (30) participants with unequal outcome (18 and 12), the rest of the columns for the second sample (unsuccessful group) we filled with zeros to make a balanced data. The successful identification is 1 and unsuccessful identification is 0 but to avoid the insignificant values for unsuccessful identification, we represent both successful and unsuccessful identifications as 1 each but separating them into 2 groups.

Table 8.2 Identification of Sections

User	Successful identification	Unsuccessful identification
1	1	1
2	1	1
3	1	1
4	1	1
5	1	1
6	1	1
7	1	1
8	1	1
9	1	1
10	1	1
11	1	1
12	1	1
13	1	0
14	1	0
15	1	0
16	1	0

17	1	0
18	1	0

To test for the significance in the outcomes of the 2 groups above, the t-test statistical test was used. T-test compares the mean and standard deviation of two samples to prove if there is any significant difference between them (SISA 2020). The t-test is calculated using the formula in equation (25) below.

$$t = \frac{(x1 - x2)}{\sqrt{\frac{s1^2}{n1} + \frac{s2^2}{n2}}} \qquad (25)$$

Where:
x1 is the mean of sample 1
s1 is the standard deviation of sample 1
n1 is the sample size of sample 1
x2 is the mean of sample 2
s2 is the standard deviation of sample 2
n**2** is the sample size in sample 2

In any significance test, there are two (2) possible hypothesis (Kim, 2019):

Null Hypothesis:	**Alternative Hypothesis:**
"There is not a significant difference between the two outcomes; any observed differences may be due to chance and sampling error''.	"There is a significant difference between the two outcomes; observed differences are most likely not due to chance or sampling error".

The probability, P, showing the significance between the two groups/samples is usually calculated. P value of 0.05 or less is significant in which case we reject the null hypothesis (accept the alternative hypothesis). If the P value is greater than 0.05, the null hypothesis is accepted and conclude that there is no significant difference between the two groups (Kim, 2019).

The computation was run on Excel sheet using tails 2 (two-tailed distribution) and type-2 (two sample equal variance), because we are not measuring the same samples at two different points in time (before and after treatment). From the computations, we obtained p=0.002571.

This means that we accept the alternative hypothesis and conclude that there is significant difference between the two outcomes. The observed differences are not due to chance or sampling error. This means that the models evaluated are effective for the task.

8.7.2 Summarisation

From the identified class in section 8.4.1, the result (identified text) was passed to the *summariser* which reduces the text to a summarised form. From our experimentation in chapter 7, the TextRank summariser was the best performing summarisation approach and was, therefore, used for the evaluation. From the completed responses by the users, the users (on average) are satisfied with the system. Table 8.2 below shows the various responses. A scale of 1-5 was used for the satisfaction score, with 5 being the highest and 1 being the lowest.

Table 8.3 Summarisation Satisfaction

Scale	Number of responses
1	1
2	5
3	13
4	10
5	1
Total	30

8.8 Statistical Test

To test the significance of the responses, we used One-Sample T-Test. This is a member of t-test family (Ross and Wilson, 2017). This statistical procedure is used to test the mean difference of the sample and the population mean, i.e. whether a population mean is significantly different from the hypothesised value. One-Sample T-test is given by the formula below.

$$t = \frac{(\mu - M)}{\frac{S}{\sqrt{n}}} \qquad (26)$$

Where:
μ is the population mean
M is the hypothesised mean
S is the standard deviation
n is the number of the observation

To test whether the population mean is greater than hypothesised mean, we test the following hypothesis:

$H0: \mu \leq M$

$H1: \mu > M$

The computation was performed at α=0.05, M=3. The result of the one-sample t-test is shown in fig. 8.1.

Mean	15.5
Variance	68.03448276
Observations	30
Pooled Variance	65.76666667
Hypothesized Mean	3
df	30
t Stat	2.110608748
P(T<=t) one-tail	0.021625367
t Critical one-tail	1.697260887
P(T<=t) two-tail	0.043250735
t Critical two-tail	2.042272456

Fig. 8.1 result of one-sample t-test

Rejection Region:

Reject H_0 if
t > 1.70 (t critical) OR
p-value ≤ α
Accept H_0 if otherwise.

Test Statistics: t = 2.11
P-value: P= 0.0216

Decision:
Since t=2.11 >1.70 OR
p-value = 0.02 < 0.05

Hence, we reject the H_0

Therefore, there is enough evidence to infer that the mean satisfaction by the users is greater than average (3). That is, users are satisfied with the outcome of the research by 3 or more levels (on a scale of 5). Hence, the outcome of the research is significant.

8.9 Future Work

In this work, we have successfully developed and demonstrated a novel approach that enables the extraction of the relevant information from the scientific research documents (publications). The approach navigates the various sections of the documents and identifies/extracts the relevant information for use in analysis. In future, the development of strategies that would combine and harmonise all the extracted data from each individual study to form a standard report. We envisage that the approach that would be able to draft a comprehensive report as a new research direction. This would combine the machine learning as well as the automatic summarisation approaches, including the meta-analysis as well as semantic techniques. In particular, the abstractive summarisation would be greatly explored.

In this work, the training only involved six thousand (6,000) full text (SRDs). In the future however, tens of thousands more SRDs could be added to provide more data for the training. Similarly, the N-gram feature was taken at the default value of 1 (1-gram), exploring higher N-grams such as 2-gram and 3-gram woud improve the models. Hence, the future work would consider higher N-grams.

Also, following on the successful development of this novel information retrieval approach with the use of machine learning and summarisation techniques, the approach will utilise, refine and generalise the developed novel methods for information retrieval from text to voice and vice versa. This way, the research shifts to more than text. This because other forms of data such as video and audio would also be integrated into the project.

9 Conclusion

The previous chapters 1-8, contained the detail of the various tasks and achievements associated with the research. This chapter being the final chapter, presents the overall conclusion of the thesis, focusing more on the main contribution of the project. This chapter also highlighted some future research direction in the domain.

9.1 Summary of Contribution

The task of extracting information (data) from the vast volumes of unstructured data, such as the SRPs in SLR, is enormous and error prone and a time-consuming task. Computer supported approach is significantly a good way to reduce the workload and enhance the process. This has produced a novel approach for information extraction from unstructured data. The approach is based on the structure of the SRPs. The scientific research publications are structured with various information about the research contained in various sections of the document. Exploring and deploying techniques based on this structure is important in addressing the challenge of information (data) extraction. Therefore, the data extraction stage of the SRL would leverage on our novel framework to reduce the time and errors and workload in conducting the review task.

In developing the approach, we divided the task into chapters, each addressing a specific goal of the research. In chapter one, the background of the research as well as the research questions were given. The aim and the objectives of the research were also specified in this chapter. In chapter two, a rigorous review of the literature on the state of science was conducted. This included the various support tools available, as well as their various degree of automation. The technologies supporting such tools and their degree of automation, including the strategies used by tools was also explored. In the end, the research gap was established, which ensured the novelty of the research was good enough for the PhD research. In chapter three (3), a three (3) layered approach was developed as the methodology for carrying out the research. This included the research design, as well as the various experiments and solutions for the project. In chapter four (4), a canonical model was produced which reflects the structure of the scientific research publication as basis for extracting the various desired data from the desired section. An algorithm was also developed in this chapter. The algorithm identifies the various sections in a document by identifying the border between sections. The canonical model was intended to be used by the system to make the extraction of the data. However, the model may not be understood by the systems, hence, artificial intelligence procedures were needed to enable the systems to understand the model. Chapter five (5) contained the machine learning model developed to achieve such intelligence of understanding the material (canonical model). Implemented as text classification task, the machine learning models achieved a good accuracy. However, same problem was implemented in a deep learning neural network. The results improved significantly. The machine learning and deep learning picks the desired section

containing the data. The section is then summarised. Appropriate summarisation techniques were selected for the task.

In general, the thesis proposed a novel framework, a unified approach to data extraction from the scientific research publications in software engineering domain. By taking advantage of the structure of the papers, machine learning (including deep learning) models were developed to identify, categorise and extract the most important bits of information from the unstructured data in SRPs. Shallow (machine learning) methods such as the Naïve Bayes, SVM, and Logistic Regression and Random Forest algorithms were used. They achieved accuracies of over 76% accuracies with over 80% precision. CNN was used for the deep learning and achieved a better performance than the machine learning methods. CNN achieved a training accuracy of 92% and validation accuracy of 85%. For the summarisation task, the *TextRank* summarisation method was used. It generated a machine summary which the users appreciated.

The results showed that hypothesis and research questions have been answered in the thesis. The set objectives are achieved, for example chapter 2 discussed the state-of-the-art approaches and clearly identified the gap in the literature. Secondly, the canonical model of the scientific document structure has been developed and evaluated. This was, in part, achieved through the document analysis algorithm discussed in section 4.4. Third, the research has produced the machine learning models capable of understanding and working through the canonical model to enable the information extraction via the structure. Fourth, the human experts were also involved. They labelled the dataset and evaluated the machine learning models developed. Also, the various automatic text summarisation approaches were appraised, and the best was selected and used on this research dataset. A very good summary, as evaluated using the state-of-the-art standard (ROGUE), was generated by the selected approach.

The research has also answered the research questions have been answered. The first research question is the possibility of automatic data extraction from the scientific research documents. The evaluation showed that most users were satisfied with the system and its outputs so the answer to the above research question is “yes”. Second, what unified approach is suitable for automatic data extraction from the scientific research documents. The research has produced a novel framework based on the canonical model of scientific document structure. Organising the contents around this structure facilitates the extraction and summarisation of the contents. The canonical model forms the kernel of the framework on which various tools and methods can be built. Third, is exploring the extent to which text and data mining technology could be

applied and developed to discover information relevant to a literature review from a research publication. This research and the results presented in Chapters 5, 6, 7 and 8 show that text and data mining tools are highly applicable to the task and are scalable for any similar task.

In conclusion, this research project addressed the most challenging task in the systematic review of software engineering literature. An efficient framework was the solution this project developed. It employed state-of-the-art technology, such as the text and data mining technologies, for the task. Most research efforts to date have been in the area of biomedicine so this project offered an interesting insight to a new domain, the software engineering domain. In this way the area was taken forward, breaking new ground.

9.1.1 Practical Application

Many tasks require the extraction of information from vast volume of documents for taking informed business decisions. For example, in judiciary, a judge or solicitor may want to obtain past judgements on some related cases. Using methods such the ones in this research, quick and accurate information could be obtained. Similarly, the ability to obtain and gather the data about drugs would be important in trying a new drug on a new ailment. During the earlier days of the COVID-19 pandemic, collecting information (reviews, efficacy) on several potential drugs/treatments would have saved the time and energy in finding an effective drug or treatment for COVID-19. Similarly, gathering evidence for any large-scale industrial projects (in all scientific disciplines) would require methods/approaches for quick and effective data extraction for use in decision making.

In a nutshell, the approach developed in this research are effective and applicable in academia as well as industry projects where information (data) from previous (similar) projects is needed for the decision making in the new project.

9.2 Limitation

The SLR has several stages involving many activities, and in several domains (areas of endeavour). This research addressed only one of the major challenging stages of the SLR process: the data extraction stage. The framework and the associated procedures produced by this research only identified and extracted the data relevant data only. The limitation of this research being that the meta-analysis is not covered (by this research). There is existing computer support for meta-analysis some of which were citied in this work. Together with what

this research has produced, the desired data can be extracted using the framework as well as performing the meta-analysis. This would ensure a more integrated platform for performing SLR assisted by the tool (automation). Similarly, combining the different findings from all the primary studies and putting them together to generate the SLR report is also not covered. Combining the respective summarised findings from each primary study to generate one overview as the SLR output is also a limitation of this research.

References

Abilio, R., Morais, F., Vale, G., Oliveira, C., Pereira, D. and Costa, H. (2015). 'Applying Information Retrieval Techniques to Detect Duplicates and to Rank References in the Preliminary Phases of Systematic: Literature Reviews'. *CLEI Electronic Journal* 18 (2), 3-3.

Abilio, R., Vale, G., Pereira, D., Oliveira, C., Morais, F. and Costa, H. (eds.) (2014). *Computing Conference (CLEI), 2014 XL Latin American*. 'Systematic Literature Review Supported by Information Retrieval Techniques: A Case Study': IEEE.

Adobe A. (2018). PDF. Three Letters that Changed the World. [Online] available from <https://acrobat.adobe.com/uk/en/acrobat/aboutadobe-pdf.html> [Jan/30 2018].

Ahuja, Y. and Yadav, S. K. (2012). Multiclass classification and support vector machine. *Global Journal of Computer Science and Technology Interdisciplinary*, *12*(11), pp.14-20. *Medicine* 4 (1), 2-7.

Akash Gupta, Harsh Sahu, Nihal Nanecha, Pradeep Kumar, Partha Pratim Roy & Victor Chang (2019): Enhancing text using emotion detected from EEG signals. *Journal of Grid Computing (2019)*, *17*(2), 325-340

Akuma, S., **Iqbal, R**. Jayne, C., Doctor, F., (2016): "Comparative analysis of relevance feedback methods based on two user studies", Computers in Human Behaviour, Vol. 60, pp. 138-146, Elsevier. (Impact Factor: 2.694, Q1).

Akuma, S., **Iqbal, R**. (2018): "Development of Relevance Feedback System using Regression Predictive Model and TF-IDF Algorithm", International Journal of Education and Management Engineering Vol. 4, pp. 31-49, MECS.

Alhabashneh, O., **Iqbal, R**., Doctor, F., James, A., (2017): "Fuzzy Rule Based Profiling Approach For Enterprise Information Seeking and Retrieval", Volumes 394–395, Pages 18–37, Information Sciences, Elsevier. doi.org/10.1016/j.ins.2016.12.040

Allahyari, M., Pouriyeh, S., Assefi, M., Safaei, S., Trippe, E. D., Gutierrez, J. B. and Kochut, K. (2017). A brief survey of text mining: Classification, clustering and extraction techniques. *arXiv preprint arXiv:1707.02919*.

Aliyu, M. B., Iqbal, R. and James, A. (2018, October). The Canonical Model of Structure for Data Extraction in Systematic Reviews of Scientific Research Articles. In *2018 Fifth International*

Conference on Social Networks Analysis, Management and Security (SNAMS) (pp. 264-271). IEEE.

Allen, C. and Richmond, K. (2011). 'The Cochrane Collaboration: International Activity within Cochrane Review Groups in the First Decade of the twenty-first Century'. *Journal of Evidence-Based.*

Al-Sallal, M., **Iqbal, R**., Palade, V., Amin, S., Chang, V., (2019): "An integrated approach for intrinsic plagiarism detection", Future Generation Computer Systems. Elsevier, Volume 96, pp. 700-712 (Impact Factor: 6.125, Q1).

Amin, S., Hijji, M., Iqbal, R., Harrop, W., Chang, V. (2019): "Fuzzy expert system-based framework for flood management in Saudi Arabia", Journal of Cluster Computing, Springer, Volume 5, Issue 2, pp. 11723–11740

Ananiadou, S., Rea, B., Okazaki, N., Procter, R. and Thomas, J. (2009). 'Supporting Systematic Reviews using Text Mining'. *Social Science Computer Review* 27 (4), 509-523.

Anthony, L. (1999). 'Writing Research Article Introductions in Software Engineering: How Accurate is a Standard Model?' *IEEE Transactions on Professional Communication* 42 (1), 38-46.

Aries, A. and Hidouci, W. K. (2019). 'Automatic Text Summarisation: What has been done and what has to be done'. *ArXiv Preprint arXiv:1904.00688*.

Arnaoudova, V., Haiduc, S., Marcus, A. and Antoniol, G. (eds.) (2015). *Software Engineering (ICSE), 2015 IEEE/ACM 37th IEEE International Conference on*. 'The use of Text Retrieval and Natural Language Processing in Software Engineering': IEEE.

Ayodele, T. O. (2010). 'Types of Machine Learning Algorithms'. *New Advances in Machine Learning*. Ed. by Anon: InTech.

Babineau, J. (2014). 'Product Review: Covidence (Systematic Review Software)'. *Journal of the Canadian Health Libraries Association/Journal De l'Association Des Bibliothèques De La Santé Du Canada* 35 (2), 68-71.

Balkissoon, D. (2017). ***Secondary Research*** [online] available from <http://designresearchtechniques.com/casestudies/secondary-research/> [November/08 2017].

Ball, R., Toh, S., Nolan, J., Haynes, K., Forshee, R. and Botsis, T. (2018). Evaluating automated approaches to anaphylaxis case classification using unstructured data from the FDA Sentinel System. *Pharmacoepidemiology and drug safety*, *27*(10), pp.1077-1084.

Barn, B., Raimondi, F., Athiappan, L. and Clark, T. (2014). 'Slrtool: A Tool to Support Collaborative Systematic Literature Reviews'.

Bates, C. (2011). *The Structure, Format, Content, and Style of a Journal-Style Scientific Paper*. Report edn.

Beller, E., Clark, J., Tsafnat, G., Adams, C., Diehl, H., Lund, H., Ouzzani, M., Thayer, K., Thomas, J. and Turner, T. (2018). 'Making Progress with the Automation of Systematic Reviews: Principles of the International Collaboration for the Automation of Systematic Reviews (ICASR)'. *Systematic Reviews* 7 (1), 77.

Beal, V. (2019). Unstructured Data [online] available from https://www.webopedia.com/TERM/U/unstructured_data.html accessed June 2019.

Bengio, Y., Courville, A. and Goodfellow, I. J. (2016). 'Deep Learning: Adaptive Computation and Machine Learning'. *Bengio.A.Courville.*

Birek, L., Grzywaczewski, A., Iqbal, R., Doctor, F., Chang, V., (2018): "A novel Big Data analytics and intelligent technique to predict driver's intent", Volume 99, pp. 226-240, Journal of Computers in Industry Elsevier. (Impact Factor: 2.85, Q1).

Brin, S. and Page, L. (2012). Reprint of: The anatomy of a large-scale hypertextual web search engine. *Computer networks*, *56*(18), pp.3825-3833.

Boruch, R., Bullock, M., Cheek, D., Cooper, H., Davies, P., McCord, J., Soydan, H., Thomas, H. and de Moya, D. (eds.) (2001). *Third International Inter-Disciplinary Evidence-Based Policies and Indicator Systems Conference*. 'The Campbell Collaboration: Concept, Status, and Plans'.

Bosco, F., Uggerslev, K., and Steel, P. (eds.) (2014). *2014 IEEE International Conference on Big Data (Big Data)*. 'Scientific Findings as Big Data for Research Synthesis: The metaBUS Project': IEEE.

Boudin, F., Nie, J., Bartlett, J. C., Grad, R., Pluye, P. and Dawes, M. (2010). 'Combining Classifiers for Robust PICO Element Detection'. *BMC Medical Informatics and Decision Making* 10 (1), 29.

Bowes, D., Hall, T. and Beecham, S. (eds.) (2012). *Proceedings of the 2nd International Workshop on Evidential Assessment of Software Technologies*. 'SLuRp: A Tool to Help Large Complex Systematic Literature Reviews Deliver Valid and Rigorous Results': ACM.

Barzilay, R. and Elhadad, M. (1999). Using lexical chains for text summarisation. *Advances in automatic text summarisation*, pp.111-121.

Brereton, P., Kitchenham, B. A., Budgen, D., Turner, M. and Khalil, M. (2007). 'Lessons from Applying the Systematic Literature Review Process within the Software Engineering Domain'. *Journal of Systems and Software* 80 (4), 571-583.

Briner, R. B. and Denyer, D. (2012). 'Systematic Review and Evidence Synthesis as a Practice and Scholarship Tool'. *Handbook of Evidence-Based Management: Companies, Classrooms and Research*, 112-129.

Brownlee, J. (2019). Bagging and Random Forest Ensemble Algorithms for Machine Learning, 22 April 2016.

Budgen, D., Turner, M., Brereton, P. and Kitchenham, B. A. (eds.) (2008). *Ppig*. 'Using Mapping Studies in Software Engineering'.

Budgen, D., Kitchenham, B., Charters, S. M., Turner, M., Brereton, P. and Linkman, S. G. (eds.) (2007). *Ease*. 'Preliminary Results of a Study of the Completeness and Clarity of Structured Abstracts'.

Budgen, D. and Brereton, P. (eds.) (2006). *Proceedings of the 28th International Conference on Software Engineering*. 'Performing Systematic Literature Reviews in Software Engineering': ACM.

Budgen, D., Charters, S., Turner, M., Brereton, P., Kitchenham, B. and Linkman, S. (eds.) (2006). *Proceedings of the 2006 International Workshop on Workshop on Interdisciplinary Software Engineering Research*. 'Investigating the Applicability of the Evidence-Based Paradigm to Software Engineering': ACM.

Cambria, E. and White, B. (2014). 'Jumping NLP Curves: A Review of Natural Language Processing Research'. *IEEE Computational Intelligence Magazine* 9 (2), 48-57.

Cant, R. P. and Cooper, S. J. (2010). 'Simulation-based Learning in Nurse Education: Systematic Review'. *Journal of Advanced Nursing* 66 (1), 3-15.

Chang, M., Chang, M., Reed, J. Z., Milward, D., Xu, J. J. and Cornell, W. D. (2016). 'Developing Timely Insights into Comparative Effectiveness Research with a Text-Mining Pipeline'. *Drug Discovery Today* 21 (3), 473-480.

Chung, G. Y. (2009). 'Towards Identifying Intervention Arms in Randomised Controlled Trials: Extracting Coordinating Constructions'. *Journal of Biomedical Informatics* 42 (5), 790-800.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine learning*, *20*(3), pp.273-297.

Cooper, I. D. (2016). 'What is a "Mapping Study?"' *Journal of the Medical Library Association: JMLA* 104 (1), 76-78.

Cruzes, D., Basili, V., Shull, F. and Jino M. (eds.), (2007). Automated Information Extraction from Empirical Software Engineering Literature: Is that Possible ESEM? First International Symposium on Empirical Software Engineering and Measurement. IEEE, 2007.

Dauphin, Y. N., Fan, A., Auli, M. and Grangier, D. (2017). Language modelling with gated convolutional networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70* (pp. 933-941). JMLR. org.

De Bruijn, B., Carini, S., Kiritchenko, S., Martin, J. and Sim, I. (2008). 'Automated Information Extraction of Key Trial Design Elements from Clinical Trial Publications'. *AMIA ...Annual Symposium Proceedings. AMIA Symposium*, 141-145.

De Oliveira, M. F. and Levkowitz, H. (2003). 'From Visual Data Exploration to Visual Data Mining: A Survey'. *IEEE Transactions on Visualisation and Computer Graphics* 9 (3), 378-394.

Dyba, T., Bergersen, G. Rye, and Sjoberg Dag, I. K. (2016). '**Evidence-Based Software Engineering**'. *Perspective on Data Science for Software Engineering*. Ed. by Anon, 149-150,151,152,153.

Dyba, T., Kitchenham, B. A. and Jorgensen, M. (2005). 'Evidence-Based Software Engineering for Practitioners'. *IEEE Software* 22 (1), 58-65.

Dybå, T. and Dingsøyr, T. (eds.) (2008). *Proceedings of the Second ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*. 'Strength of Evidence in Systematic Reviews in Software Engineering': ACM.

EFSA Guidance for those carrying out systematic reviews, European Food Safety Authority (2010). 'Application of Systematic Review Methodology to Food and Feed Safety Assessments to Support Decision Making'. *EFSA Journal* 8 (6), 1637.

Erkan, G. and Radev, D. R. (2004). LexRank: Graph-based lexical centrality as salience in text summarisation. *Journal of artificial intelligence research*, *22*, pp.457-479.

EPPI-Centre (2019). *EPPI-Centre* [online] available from <https://eppi.ioe.ac.uk/cms/> [July 2018 2019].

Felizardo, K. R., MacDonell, S. G., Mendes, E. and Maldonado, J. C. (2012). 'A Systematic Mapping on the use of Visual Data Mining to Support the Conduct of Systematic Literature Reviews'.

Fellbaum, C. (2010). 'Wordnet'. *Theory and Applications of Ontology: Computer Applications*. Ed. by Anon: Springer, 231-243.

Fernández-Sáez, A. M., Bocco, M. G. and Romero, F. P. (eds.) (2010). *Icsoft (2)*. 'SLR-Tool: A Tool for Performing Systematic Literature Reviews'.

Fischer, B. A. and Zigmond, M. J. (2004). 'Components of a Research Article'. *2007-02-24].Http://www.Soudoc.com/bbs/simple/index.Php.*

Gandomi, A. and Haider, M. (2015). 'Beyond the Hype: Big Data Concepts, Methods, and Analytics'. *International Journal of Information Management* 35 (2), 137-144.

Gupta, V. and Lehal, G. S. (2010). A survey of text summarisation extractive techniques. *Journal of emerging technologies in web intelligence*, *2*(3), pp.258-268.

Guyatt, G., Cairns, J., Churchill, D., Cook, D., Haynes, B., Hirsh, J., Irvine, J., Levine, M., Levine, M. and Nishikawa, J. (1992). 'Evidence-Based Medicine: A New Approach to Teaching the Practice of Medicine'. *Jama* 268 (17), 2420-2425.

Greenhalgh, T. and Peacock, R. 2005. Effectiveness and efficiency of search methods in systematic reviews of complex evidence: audit of primary sources. *Bmj*, *331*(7524), pp.1064-1065.

Grzywaczewski, A., **Iqbal, R**., James, A., Halloran, J., (2009): "An Investigation of User Behaviour Consistency for Context-Aware Information Retrieval Systems ", International Journal of Advanced Pervasive and Ubiquitous Computing (IJAPUC), 1(4), pp. 69-90 (http://new.igi-global.com/Bookstore/Article.aspx?TitleId=41705)

Grzywaczewski, A., and **Iqbal, R**., (2011): "Software engineers' information behaviour and implicit relevance indicators", Journal of Knowledge and Web Intelligence, Inderscience, Volume 2, Issue 2/3, pp., 185-201.

Grzywaczewski, A., and **Iqbal, R**., (2012):"Task-Specific Information Retrieval Systems for Software Engineers", Journal of Computer and System Sciences, Elsevier, Volume 78, Issue 4, pp., 1204-1218. (Impact Factor: 1.138)

Haddaway, N. R. and Pullin, A. S. (2014). 'The Policy Role of Systematic Reviews: Past, Present and Future'. *Springer Science Reviews* 2 (1-2), 179-183.

HakemZadeh, F. (2012). 'An Introduction to Systematic Reviews London, UK: Sage Publications, (2012). 304 Pp. ISBN-9781849201803'. *Canadian Journal of Administrative Sciences/Revue Canadienne Des Sciences De l'Administration* 29 (4), 378-379.

Hara, K. and Matsumoto, Y. (2007). 'Extracting Clinical Trial Design Information from MEDLINE Abstracts'. *New Generation Computing* 25 (3), 263-275.

Harrington, P. (2012). *Machine learning in action*. Manning Publications Co.

Hashimoto, K., Kontonatsios, G., Miwa, M. and Ananiadou, S. (2016). 'Topic Detection using Paragraph Vectors to Support Active Learning in Systematic Reviews'. *Journal of Biomedical Informatics* 62, 59-65.

Hassani, H. (2017). 'Research Methods in Computer Science: The Challenges and Issues'. *ArXiv Preprint arXiv: 1703.04080*.

Hassanzadeh, H., Groza, T. and Hunter, J. (2014). 'Identifying Scientific Artefacts in Biomedical Literature: The Evidence Based Medicine use Case'. *Journal of Biomedical Informatics* 49, 159-170.

Hearst, M. (2003). 'What is Text Mining?' *SIMS, UC Berkeley*.

Hernandes, E., Zamboni, A., Fabbri, S. and Thommazo, A. D. (2012). 'Using GQM and TAM to Evaluate StArt-a Tool that Supports Systematic Review'. *CLEI Electronic Journal* 15 (1), 3-3.

Hevner, A. and Chatterjee, S. (2010). 'Design Science Research in Information Systems'. *Design Research in Information Systems*. Ed. by Anon: Springer, 9-22.

Hossin, M. and Sulaiman, M. N. (2015). A review on evaluation metrics for data classification evaluations. *International Journal of Data Mining & Knowledge Management Process*, *5*(2), p.1.

Hox, J. J. and Boeije, H. R. (2005). 'Data Collection, Primary Versus Secondary'.

Hsu, W., Speier, W. and Taira, R. K. 2012. Automated extraction of reported statistical analyses: towards a logical representation of clinical trial literature. *AMIA Annual Symposium Proceedings* (Vol. 2012, p. 350). American Medical Informatics Association.

Huang, K., Chiang, I., Xiao, F., Liao, C., Liu, C. C. and Wong, J. (2013). 'PICO Element Detection in Medical Text without Metadata: Are First Sentences enough?' *Journal of Biomedical Informatics* 46 (5), 940-946.

Huang, K., Liu, C. C., Yang, S., Xiao, F., Wong, J., Liao, C. and Chiang, I. (eds.) (2011). *Granular Computing (GrC), 2011 IEEE International Conference*. 'Classification of PICO Elements by Text Features Systematically Extracted from PubMed Abstracts': IEEE.

Ikonomakis, M., Kotsiantis, S. and Tampakas, V. (2005). 'Text Classification using Machine Learning Techniques'. *WSEAS Transactions on Computers* 4 (8), 966-974.

Iqbal, K, Odetayo M. James, A., Iqbal, R., Kumar, N., Bama S.,(2015): “An efficient image retrieval scheme for colour enhancement of embedded and distributed surveillance images”, Journal of Neurocomputing, Vol 174, Part 2, pp. 413-430

Iqbal, R., Grzywaczewski, A., Halloran, J., Doctor F., Iqbal, K., (2017): “Design implications for task-specific search utilities for retrieval and reengineering of code”, Enterprise Information Systems, Vol 11, Issue 5, pp. 738–757, Taylor and Francis, pp 1751-7575. doi: 10.1080/17517575.2015.1086494 (Impact Factor: 2.269, Q1)

Iqbal, R., Maniak, T., Doctor, F., Karyotis, C., (2019): “Fault detection and isolation in industrial processes using deep learning approaches”, IEEE Transaction on Industrial Informatics, Volume 15, Issue 5, pp 3077-2084 (Impact Factor: 5.43, Q1)

JabRef (2019). *Jabref* [online] available from <http://www.jabref.org/> [January/2018 2018].

James, K. L., Randall, N. P. and Haddaway, N. R. (2016). 'A Methodology for Systematic Mapping in Environmental Sciences'. *Environmental Evidence* 5 (1), 7.

Jaspers, S., De Troyer, E. and Aerts, M. (2018). 'Machine Learning Techniques for the Automation of Literature Reviews and Systematic Reviews in EFSA'. *EFSA Supporting Publications* 15 (6), 1427E.

Javaid, N. (2018). ***Machine Learning — Text Processing*** [online] available from <https://towardsdatascience.com/machine-learning-text-processing-1d5a2d638958> [October 2018 2018].

Joachims, T. (1998). Text categorisation with support vector machines: Learning with many relevant features. In *European conference on machine learning* (pp. 137-142). Springer, Berlin, Heidelberg.

Jonnalagadda, S. R., Goyal, P. and Huffman, M. D. (2015). 'Automating Data Extraction in Systematic Reviews: A Systematic Review'. *Systematic Reviews* 4 (1), 78.

Jordan, M. I. and Mitchell, T. M. (2015). 'Machine Learning: Trends, Perspectives, and Prospects'. *Science (New York, N.Y.)* 349 (6245), 255-260.

Kalchbrenner, N., Grefenstette, E. and Blunsom, P. (2014). A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*.

Karyotis, C., Doctor, F., **Iqbal, R**., James, A., Cheng, V., (2017): "A fuzzy computational model of emotion for cloud based sentiment analysis", Information Sciences, Elsevier.

Keele, S. (2007). 'Guidelines for Performing Systematic Literature Reviews in Software Engineering'. *Technical Report, Ver. 2.3 EBSE Technical Report. EBSE*. Ed. by Anon: sn.

Kelly, C. and Yang, H. 2013. A system for extracting study design parameters from nutritional genomics abstracts. *Journal of integrative bioinformatics*, *10*(2), pp.82-93.

Khan, A., Baharudin, B., Lee, L. H. and Khan, K. (2010). 'A Review of Machine Learning Algorithms for Text-Documents Classification'. *Journal of Advances in Information Technology* 1 (1), 4-20.

Khatri, C., Singh, G. and Parikh, N. (2018). Abstractive and Extractive Text Summarisation using Document Context Vector and Recurrent Neural Networks. *arXiv preprint arXiv:1807.08000*.

Kim, S. N., Martinez, D., Cavedon, L. and Yencken, L. (2011). 'Automatic Classification of Sentences to Support Evidence Based Medicine'. *BMC Bioinformatics* 12 (2), S5.

Kim, Y. (2014). Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Kiritchenko, S., De Bruijn, B., Carini, S., Martin, J. and Sim, I. (2010). 'ExaCT: Automatic Extraction of Clinical Trial Characteristics from Journal Publications'. *BMC Medical Informatics and Decision Making* 10 (1), 56.

Kitchenham, B. and Brereton, P. (2013). 'A Systematic Review of Systematic Review Process Research in Software Engineering'. *Information and Software Technology* 55 (12), 2049-2075.

Kitchenham, B., Brereton, O. P., Budgen, D., Turner, M., Bailey, J. and Linkman, S. (2009). 'Systematic Literature Reviews in Software engineering–a Systematic Literature Review'. *Information and Software Technology* 51 (1), 7-15.

Kitchenham, B. (2004). 'Procedures for Performing Systematic Reviews'. *Keele, UK, Keele University* 33 (2004), 1-26.

Kitchenham, B. A., Dyba, T. and Jorgensen, M. (eds.) (2004). *Proceedings of the 26th International Conference on Software Engineering*. 'Evidence-Based Software Engineering': IEEE Computer Society.

Kohl, C., Craig, W., Frampton, G., Garcia-Yi, J., van Herck, K., Kleter, G. A., Krogh, P. H., Meissle, M., Romeis, J. and Spök, A. (eds.) (2013). *GMOs in Integrated Plant Production*. 'Developing a Good Practice for the Review of Evidence Relevant to GMO Risk Assessment'.

Kleinberg, J. M. (1998). Authoritative sources in a hyperlinked environment. *In Proceedings of the ACM-SIAM symposium on discrete algorithms*.

Kumar, A., Dabas, V. and Hooda, P. (2018). Text classification algorithms for mining unstructured data: a SWOT analysis. *International Journal of Information Technology*, pp.1-11.

Landhuis, E. (2016). Scientific literature: information overload. *Nature*, *535*(7612), pp.457-458.

Leopold, E. and Kindermann, J. (2002). 'Text Categorisation with Support Vector Machines. How to Represent Texts in Input Space' *Machine Learning* 46 (1-3), 423-444.

Liakata, M., Saha, S., Dobnik, S., Batchelor, C. and Rebholz-Schuhmann, D. (2012). 'Automatic Recognition of Conceptualisation Zones in Scientific Articles and Two Life Science Applications'. *Bioinformatics* 28 (7), 991-1000.

Lin, C. (ed.) (2004). *Text Summarisation Branches Out*. 'Rouge: A Package for Automatic Evaluation of Summaries'.

Lin, S., Ng, J., Pradhan, S., Shah, J., Pietrobon, R. and Kan, M. (eds.) (2010). *Proceedings of the NAACL HLT 2010 Second Louhi Workshop on Text and Data Mining of Health Documents*. 'Extracting Formulaic and Free Text Clinical Research Articles Metadata using Conditional Random Fields': Association for Computational Linguistics.

Iosad, P. (2017). *Substance-free Framework for Phonology: agl*. Edinburgh University Press.

Luhn, H. P. (1958). The automatic creation of literature abstracts. *IBM Journal of research and development*, *2*(2), pp.159-165.

Madsen, R. E., Sigurdsson, S., Hansen, L. K. and Larsen, J. (eds.) (2004). *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*. 'Pruning the Vocabulary for Better Context Recognition': IEEE.

Urra Medina, E. and Barría Pailaquilén, R. M. (2010). Systematic review and its relationship with evidence-based practice in health. *Revista latino-americana de enfermagem*, *18*(4), pp.824-831.

Malheiros, V., Hohn, E., Pinho, R. and Mendonca, M. (eds.) (2007). *Empirical Software Engineering and Measurement, 2007. ESEM 2007. First International Symposium on*. 'A Visual Text Mining Approach for Systematic Reviews': IEEE.

Mani, I. (2001). *Automatic Summarisation*. John Benjamins Publishing.

Manning, C. D., Raghavan, P. and Schütze, P. (2009). *An Introduction to Information Retrieval*. Cambridge: Cambridge University Press.

Marchall, C. and Brereton, O. (2015). 'Systematic Review Toolbox: A Catalogue of Tools to Support Systematic Reviews'.

Marr, B. (2018). How Much Data Do We Create Every Day? The Mind-Blowing Stats Everyone Should Read [online] available from https://www.forbes.com/sites/bernardmarr/2018/05/21/how-much-data-do-we-create-every-day-the-mind-blowing-stats-everyone-should-read/ accessed April 2019.

Marshall, C. (2016). *Tool Support for Systematic Reviews in Software Engineering*.

Marshall, C. and Brereton, P. (eds.) (2015). *Proceedings of the 19th International Conference on Evaluation and Assessment in Software Engineering*. 'Systematic Review Toolbox: A Catalogue of Tools to Support Systematic Reviews': ACM.

Marshall, C., Brereton, P. and Kitchenham, B. (2014, May). Tools to support systematic reviews in software engineering: a feature analysis. *Proceedings of the 18th international conference on evaluation and assessment in software engineering* (p. 13). ACM.

Marshall, I. J., Kuiper, J. and Wallace, B. C. (2015). 'RobotReviewer: Evaluation of a System for Automatically Assessing Bias in Clinical Trials'. *Journal of the American Medical Informatics Association* 23 (1), 193-201.

Maswana, S., Kanamaru, T. and Tajino, A. (2015). 'Move Analysis of Research Articles Across Five Engineering Fields: What they Share and what they do Not'. *Ampersand* 2, 1-11.

Matters, E. (2002). *The Database of Abstracts of Reviews of Effects (DARE).United Kingdom: The University of York*.

McGowan, J. and Sampson, M. (2005). 'Systematic Reviews Need Systematic Searchers (IRP)'. *Journal of the Medical Library Association* 93 (1), 74.

McCargar, V. (2004). Statistical approaches to automatic text summarisation. *Bulletin of the American Society for Information Science and Technology*, *30*(4), pp.21-25.

Mendeley (2019). *Mendeley* [online] available from <https://www.mendeley.com/?interaction_required=true> [June/2018 2018].

Mihalcea, R. (ed.) (2004). *Proceedings of the ACL 2004 on Interactive Poster and Demonstration Sessions*. 'Graph-Based Ranking Algorithms for Sentence Extraction, Applied to Text Summarisation': Association for Computational Linguistics.

Mihalcea, R. and Tarau, P. (eds.) (2004). *Emnlp*. 'TextRank: Bringing Order into Text'.

Millard, L. A., Flach, P. A. and Higgins, J. P. (2015). 'Machine Learning to Assist Risk-of-Bias Assessments in Systematic Reviews'. *International Journal of Epidemiology* 45 (1), 266-277.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. and Dean, J.: Distributed representations of words and phrases and their compositionality. Proceedings of NIPS 2013 (2013).

Mitchell, T. M. (1999). 'Machine Learning and Data Mining'. *Communications of the ACM* 42 (11), 30-36.

Mitkov, R. (1993) 'Automatic Abstracting in a Limited Domain'.

Molléri, J. S. and Benitti, F. B. V. (eds.) (2015). *Proceedings of the 19th International Conference on Evaluation and Assessment in Software Engineering*. 'SESRA: A Web-Based Automated Tool to Support the Systematic Literature Review Process': ACM.

Moens, M. F., Uyttendaele, C. and Dumortier, J. (2000). Intelligent information extraction from legal texts. *Information & Communications Technology Law*, *9*(1), pp.17-26.

Mulrow, C. D. (1994). 'Rationale for Systematic Reviews'. *BMJ (Clinical Research Ed.)* 309 (6954), 597-599.

Nanos, A. G., James, A. E., Iqbal, R. and Hedley, Y. (eds.) (2017). *Computer Supported Cooperative Work in Design (CSCWD), 2017 IEEE 21st International Conference*. 'Content Summarisation of Conversation in the Context of Virtual Meetings: An Enhanced TextRank Approach': IEEE.

O'Mara-Eves, A., Thomas, J., McNaught, J., Miwa, M. and Ananiadou, S. (2015). 'Using Text Mining for Study Identification in Systematic Reviews: A Systematic Review of Current Approaches'. *Systematic Reviews* 4 (1), 5.

Octaviano, F., Silva, C. and Fabbri, S. (eds.) (2016). *Proceedings of the 20th International Conference on Evaluation and Assessment in Software Engineering*. 'Using the SCAS Strategy to Perform the Initial Selection of Studies in Systematic Reviews: An Experimental Study': ACM.

Ouzzani, M., Hammady, H., Fedorowicz, Z. and Elmagarmid, A. (2016). 'Rayyan—a Web and Mobile App for Systematic Reviews'. *Systematic Reviews* 5 (1), 210.

Paiva, C. E., Lima, João Paulo da Silveira Nogueira and Paiva, B. S. R. (2012). 'Articles with Short Titles Describing the Results are Cited More Often'. *Clinics* 67 (5), 509-513.

Page, L., Brin, S., Motwani, R. and Winograd, T. (1999). *The PageRank citation ranking: Bringing order to the web*. Stanford InfoLab.

Pin Ni , Yuming Li , Victor Chang **(2020): Research on Text Classification Based on Automatically Extracted Keywords. *International Journal of Enterprise Information Systems (IJEIS)* (2020), *16*(4), 1-16.**

Pollock, R. (2019). Tools for Extracting Data and Text from PDFs - A Review [online] available from https://okfnlabs.org/blog/2016/04/19/pdf-tools-extract-text-and-data-from-pdfs.html accessed June 2019.

Peffers, K., Tuunanen, T., Rothenberger, M. A. and Chatterjee, S. (2007). 'A Design Science Research Methodology for Information Systems Research'. *Journal of Management Information Systems* 24 (3), 45-77.

Perneger, T. V. and Hudelson, P. M. (2004). 'Writing a Research Article: Advice to Beginners'. *International Journal for Quality in Health Care* 16 (3), 191-192.

Petersen, K., Vakkalanka, S. and Kuzniarz, L. (2015). 'Guidelines for Conducting Systematic Mapping Studies in Software Engineering: An Update'. *Information and Software Technology* 64, 1-18.

Petticrew, M. and Roberts, H. (2008). *Systematic Reviews in the Social Sciences: A Practical Guide*: John Wiley & Sons.

Rathbone, J., Hoffmann, T. and Glasziou, P. (2015)'Faster Title and Abstract Screening? Evaluating Abstrackr, a Semi-Automated Online Screening Program for Systematic Reviewers'. *Systematic Reviews* 4 (1), 80.

Radev, D. R., Blair-Goldensohn, S. and Zhang, Z. (2001, September). Experiments in single and multidocument summarisation using MEAD. *First document understanding conference* (p. 1À8).

Rajat, H. (2018). Choosing the Right Machine Learning Algorithm. *Hackernoon*.

Rajoub, B., 2020. Supervised and unsupervised learning. In *Biomedical Signal Processing and Artificial Intelligence in Healthcare* (pp. 51-89). Academic Press.

Ranawana, R. and Palade, V. (2006, July). Optimised precision - a new measure for classifier performance evaluation. In *2006 IEEE International Conference on Evolutionary Computation* (pp. 2254-2261). IEEE.

Rappoport, N. and Shamir, R. (2018). Multi-omic and multi-view clustering algorithms: review and cancer benchmark. *Nucleic acids research*, *46*(20), pp.10546-10562.

Refworks (2019). *Refworks* [online] available from <refworks.com> [April/2017 2017].

Rennels, G. D., Shortliffe, E. H., Stockdale, F. E. and Miller, P. L. (1989). 'A Computational Model of Reasoning from the Clinical Literature'. *AI Magazine* 10 (1), 49-49.

Restificar, A. and Ananiadou, S. (eds.) (2012). *Proceedings of the ACM Sixth International Workshop on Data and Text Mining in Biomedical Informatics*. 'Inferring Appropriate Eligibility Criteria in Clinical Trial Protocols without Labelled Data': ACM.

Robert, V. Labaree (2017). *Organising Your Social Sciences Research Paper: 6. the Methodology* [online] available from <http://libguides.usc.edu/writingguide/methodology> [August/16 2017].

Robinson, D. A. (2012). *Finding Patient-Oriented Evidence in PubMed Abstracts.*

Rodriguez, M.Z., Comin, C.H., Casanova, D., Bruno, O.M., Amancio, D.R., Costa, L.D.F. and Rodrigues, F.A. (2019). Clustering algorithms: A comparative approach. *PloS one*, *14*(1), p.e0210236.

Rokaha, B., Ghale, D.P. and Gautam, B.P. (2018). Enhancement of Supermarket Business and Market Plan by Using Hierarchical Clustering and Association Mining Technique. In *2018 International Conference on Networking and Network Applications (NaNA)* (pp. 384-389). IEEE

Rowley, J. and Slack, F. (2004). 'Conducting a Literature Review'. *Management Research News* 27 (6), 31-39.

REviewER (2013). REviewERSLR available from https://sites.google.com/site/eseportal/tools/reviewer [June 2017].

Sadeghi, R. and Treglia, G. (2017). 'Systematic Reviews and Meta-Analyses of Diagnostic Studies: A Practical Guideline'. *Clinical and Translational Imaging* 5 (2), 83-87.

SAS, S. (2012). *Machine Learning* [online] available from <https://www.sas.com/en_gb/insights/analytics/machine-learning.html#> [November/27 2017].

Schütze, H., Manning, C. D. and Raghavan, P. (2008, June). Introduction to information retrieval. *Proceedings of the international communication of association for computing machinery*.

Shakeel, P.M., Baskar, S., Dhulipala, V.S. and Jaber, M.M., 2018. Cloud based framework for diagnosis of diabetes mellitus using K-means clustering. *Health information science and systems*, *6*(1), p.16.

Silge, J. and Robinson, D. (2018). Term frequency and inverse document frequency (tf-idf) using tidy data principles.*conference* (Vol. 4).

Springer (2020), Overview of IMRaD Structure. Available at https://www.springer.com/gp/authors-editors/journal-author/overview-of-imrad-structure/1408 accessed 30th June 2020.

Summerscales, R., Argamon, S., Hupert, J. and Schwartz, A. (2009). Identifying treatments, groups, and outcomes in medical abstracts. In *The Sixth Midwest Computational Linguistics Colloquium (MCLC 2009)*.

Summerscales, R. L., Argamon, S., Bai, S., Hupert, J. and Schwartz, A. (2011). November. Automatic summarisation of results from clinical trials. In *2011 IEEE International Conference on Bioinformatics and Biomedicine* (pp. 372-377). IEEE.

Song, M. H., Lee, Y. H. and Kang, U. G. (2013). 'Comparison of Machine Learning Algorithms for Classification of the Sentences in Three Clinical Practice Guidelines'. *Healthcare Informatics Research* 19 (1), 16-24.

Swales, J. (1990). *Genre Analysis: English in Academic and Research Settings*: Cambridge University Press.

Tan, A. (ed.) (1999). *Proceedings of the PAKDD 1999 Workshop on Knowledge Discovery from Advanced Databases*. 'Text Mining: The State of the Art and the Challenges'.

The Nordic Cochrane Centre, The Cochrane Collaboration (2014). *Review Manager (RevMan)* [online] available from <https://community.cochrane.org/help/tools-and-software/revman-5> [2019].

Thomas, J., Brunton, J. and Graziosi, S. (2010). 'EPPI-Reviewer 4.0: Software for Research Synthesis. EPPI-Centre Software. London: Social Science Research Unit'. *Institute of Education, University of London.*

Thomas, J., McNaught, J. and Ananiadou, S. (2011). 'Applications of Text Mining within Systematic Reviews'. *Research Synthesis Methods* 2 (1), 1-14.

Tomassetti, F., Rizzo, G., Vetro, A., Ardito, L., Torchiano, M. and Morisio, M. (eds.) (2011). *Evaluation & Assessment in Software Engineering (EASE 2011), 15th Annual Conference*. 'Linked Data Approach for Selection Process Automation in Systematic Reviews': IET.

Torres-Moreno, J. M. (ed.) (2014). *Automatic text summarisation*. John Wiley & Sons.

Tranfield, D., Denyer, D. and Smart, P. (2003). 'Towards a Methodology for Developing Evidence-Informed Management Knowledge by Means of Systematic Review'. *British Journal of Management* 14 (3), 207-222.

Trofimovich, P., Collins, L., Cardoso, W., White, J. and Horst, M. (2012). 'A Frequency-Based Approach to L2 Phonological Learning: Teacher Input and Student Output in an Intensive ESL Context'. *TESOL Quarterly* 46 (1), 176-187.

Tsafnat, G., Dunn, A., Glasziou, P. and Coiera, E. (2013). 'The Automation of Systematic Reviews'. *BMJ (Clinical Research Ed.)* 346, f139.

Urra Medina, E. and Barría Pailaquilén, R. M. (2010). 'Systematic Review and its Relationship with Evidence-Based Practice in Health'. *Revista Latino-Americana De Enfermagem* 18 (4), 824-831.

van Altena, A., Spijker, R. and Olabarriaga, S. (2019). 'Usage of Automation Tools in Systematic Reviews'. *Research Synthesis Methods* 10 (1), 72-82.

Verbeke, M., Van Asch, V., Morante, R., Frasconi, P., Daelemans, W. and De Raedt, L. (eds.) (2012). *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. 'A Statistical Relational Learning Approach to Identifying Evidence Based Medicine Categories': Association for Computational Linguistics.

Visser, E. (2010). 'Performing Systematic Literature Reviews with Researchr: Tool Demonstration'. *Technical Report Series TUD-SERG-2010-010.*

Von Alan, R. H., March, S. T., Park, J. and Ram, S. (2004). 'Design Science in Information Systems Research'. *MIS Quarterly* 28 (1), 75-105.

Vu, N. T., Adel, H., Gupta, P. and Schütze, H. (2016). Combining recurrent and convolutional neural networks for relation classification. *arXiv preprint arXiv:1605.07333*.

Wallace, B. C., Small, K., Brodley, C. E., Lau, J. and Trikalinos, T. A. (eds.) (2012). *Proceedings of the 2nd ACM SIGHIT International Health Informatics Symposium*. 'Deploying an Interactive Machine Learning System in an Evidence-Based Practice Centre: Abstrackr': ACM.

Wang, S. and Manning, C. D. (2012, July). Baselines and bigrams: Simple, good sentiment and topic classification. *Proceedings of the 50th annual meeting of the association for computational linguistics: Short papers-volume 2* (pp. 90-94). Association for Computational Linguistics.

Wohlin, C., Runeson, P., Neto, Paulo Anselmo da Mota Silveira, Engström, E., do Carmo Machado, I. and De Almeida, E. S. (2013). 'On the Reliability of Mapping Studies in Software Engineering'. *Journal of Systems and Software* 86 (10), 2594-2610.

Yeung, S., Fathi, A. and Fei-Fei, L. (2014). 'Videoset: Video Summary Evaluation through Text'. *ArXiv Preprint arXiv: 1406.5824*.

Yogan, J. K., Goh, O. S., Halizah, B., Ngo, H. C. and Puspalata, C. (2016). A review on automatic text summarisation approaches. *Journal of Computer Science*, *12*(4), pp.178-190.

Xiao, Y., Huang, C., Huang, J., Kaku, I. and Xu, Y., 2019. Optimal mathematical programming and variable neighborhood search for k-modes categorical data clustering. *Pattern Recognition*, *90*, pp.183-195.

Xu, R., Garten, Y., Supekar, K. S., Das, A. K., Altman, R. B. and Garber, A. M. 2007. Extracting subject demographic information from abstracts of randomised clinical trial reports. *Medinfo 2007: Proceedings of the 12th World Congress on Health (Medical) Informatics; Building Sustainable Health Systems* (p. 550). IOS Press.

Zhao, J., Jiang, Y. and Liu, Y. (2019). *Sentence-Level Extractive Text Summarisation*. Stanford Uni: Stanford.

Zhao, J., Bysani, P. and Kan, M. Y. (2012). 'Exploiting Classification Correlations for the Extraction of Evidence-Based Practice Information'. *AMIA ...Annual Symposium Proceedings. AMIA Symposium* 2012, 1070-1078.

Zhang, P.Y. and Li, C.H. (2009, August). Automatic text summarisation based on sentences clustering and extraction. In *2009 2nd IEEE international conference on computer science and information technology* (pp. 167-170). IEEE.

Zhou, Y., Zhang, H., Huang, X., Yang, S., Babar, M. A. and Tang, H. (eds.) (2015). *Proceedings of the 19th International Conference on Evaluation and Assessment in Software Engineering*. 'Quality Assessment of Systematic Reviews in Software Engineering: A Tertiary Study': ACM.

Zhu, H., Ni, Y., Cai, P., Qiu, Z. and Cao, F. (2012). 'Automatic Extracting of Patient-Related Attributes: Disease, Age, Gender and Race'. *Studies in Health Technology and Informatics* 180, 589-593.

Appendix
Appendix A: Instruction for the Exercise

Instruction for the participants

Please read the following instructions carefully before you begin the evaluation exercise. You can also ask for further clarifications from the primary researcher. He will be with you throughout the process.

1. The system is already setup for you. No need for any configurations or changes to any part of the system.
2. As you must have been informed in the consent form and participant information sheet, you must have brought your own piece of document to be used for the exercise. This is the preferred option. However, the primary researcher can offer an alternative arrangement if required.
3. The system evaluation is in two (2) parts. Each part works on the same piece of document.
4. Save the document on the location accessible by the tool for the evaluation. The researcher would help you put the document in the appropriate location accessible by the tool.
5. All the needed pre-processing would be done or assisted by the researcher who would be with you throughout the duration of the exercise which would last for approximately 20 minutes.
6. The tool would attempt to pick or identify two (2) sections from the paper. These are the 'methodology' and 'result' sections. This is the machine learning aspect of the system.
7. Look carefully at the output produced by the tool for accuracy or otherwise.
8. The next part of the system would attempt to summarise this identified section to give the information from that section in a concise form.
9. Rate the summary produced by the system using either the summary you produced or your intuitive judgement.
10. Complete the evaluation form you are provided with. The form is named ***Form EF1***.

If anything is unclear, feel free to ask the primary researcher for clarification or explanation.

Thank you very much for your time.

Appendix B: Participant Information Sheet

Machine Learning and Summarisation project

Participant Information Sheet

This is a formal request for your participation in an evaluation task in a project that creates a tool to support the automatic data extraction in the systematic review in scientific research articles in the software engineering domain. The tool identifies some desired sections from a document (research paper) and presents the summarised result of the identified section from the paper. We would like you to take part in our evaluation.

We want you to evaluate the accuracy of the machine learning identifications of the various sections of the paper and also rank the summary produced by the tool. We would later compare the summary output from our tool with the human summary.

No personal information is required of you, and no personal information will be given to you. All responses will be anonymised during the entire process. Your participation is anonymous and voluntary. All data from this research will be stored on the students J-drive located at Coventry University. We will not keep records of which participant makes which judgement on the summary ranking. After the study the list of participants will be destroyed.

Also, there are no personal risks or dangers in taking part, except that it will take around 20 minutes of your time.

In undertaking this research process, a due diligence to underpin ethical issues pertaining to this study is being observed. This includes the use of informed consent, maintaining confidentiality and ensuring a fair anonymity throughout the process.

Appendix C: Consent Statement

Consent statement

I have read and fully understand the attached participant information sheet and by signing below I consent to participate in this study.

I understand that I have right to withdraw from the study participation without giving reason at any time during the study period.

I also understand that all data collected in the study will be held strictly in anonymous form.

Signed by participant.....

Date.....

Signed by researcher.....

Date.....

Appendix D: Tool Evaluation Sheet

Please read the instructions carefully before completing this form EF1

Based on the performed task, complete the following evaluations carefully. There are three (3) sections in the form. The 2 corresponds to the two (2) aspects demonstrated in the system. While the 3rd section is about the tool as a whole

Part 1: The machine learning

1. On a binary scale of 0-1, how would you score the accuracy of the identified section from your document (research paper you brought or were provided), with zero being wrong identification and 1 correct identification. tick appropriately

1	Correct	
0	Wrong	

2. On a scale of 1-5, how 'easy to use' would you score the machine learning aspect of the tool. 1 being the least score and 5 the highest score. Thick appropriately

1	Poor	
2	Fair	
3	Satisfactory	
4	good	
5	very good	

Part 2: The summary

1. On a scale of 1-5, how would you rate the 'correctness' of the summary produced by the system? You use your own crafted summary or your intuition as a reference.

1	Poor	
2	Fair	
3	Satisfactory	
4	good	
5	very good	

2. Any comment regarding the system generated summary? Fill the box below

Part 3: The Value of the System:

1. Does the system add value to the SLR automation? Thick appropriately

   Yes ◯ No ◯ cannot say ◯

2. Any comment about the system or how to improve it?

Appendix E: Tools experimentation and Assessment details

To evaluate the support/automation provided by the tools, following experiment was carried out. The tools have several but different availability options. Some are freely available online or in desktop version while others are only available after payment. We downloaded and installed the desktop version locally as well as the full figurations settings.

The tool's documentation was used as guideline for assessing all the features that the tool supports. We scored each tool against each feature to produce a raw score. The raw score is a number that indicates the degree of support the tool offers.

Each tool was initially scored against each feature by the first author (CM). The scores were then discussed by all the authors to produce a set of validated raw scores. A spreadsheet was used to record raw scores, weighted scores and overall scores.

Using a single simple judgement scale (1, 05, 0) first used by Marshal *et al.,* (2014), each tool was rated against the features discussed previously. A score of 1 indicates that the feature is fully present or strongly supported. We assigned a score of 0.5 where a feature was partly present or partially supported and 0 is awarded where such is not supported at all. It is worthy to note that this score scale was only randomly chosen Marshal *et al.,* (2014).

Three (3) researchers independently carried the assessment. The result was harmonised later. Where there was a variation among the researchers, the majority score was taken. For example, if 2 researchers score 1, and the other gave a different score, score 1 is taken.

Appendix F: Automatic Summarisation.

A) TF-IDF scores

'Result': 0.35097542827641776, 'discussion': 0.35097542827641776, 'Due': 0.35097542827641776, 'WEKA': 0.35097542827641776, 'allows': 0.35097542827641776, 'change': 0.35097542827641776, 'find': 0.35097542827641776, 'appropriate': 0.35097542827641776, 'use': 0.09338552980898464, 'workbench': 0.08638161507331078, 'parameters': 0.2124705709517297, 'tested': 0.36805487239580503, 'Firstly': 0.21448498394669974, 'default': 0.21448498394669974, 'according': 0.21448498394669974, 'confusion-matrices': 0.21448498394669974, 'find': 0.21448498394669974, 'achieved': 0.21448498394669974, 'by the': 0.21448498394669974, 'algorithms': 0.1645364096634491, '"': 0.05315791696819125, 'best': 0.12183009732617887, 'results': 0.36805487239580503, 'values': 0.17263233889828036, 'respective': 0.0746518222262834, 'accuracy': 0.16496235666138034, 'classifier': 0.23955351320024115, 'was assigned': 0.1930364855520298, 'M': 0.1930364855520298, 'stop': 0.1930364855520298, 'four': 0.16993104604585726, 'two': 0.2301764518643738, 'seven': 0.0965182427760149, 'layers': 0.0965182427760149, '0.21': 0.0965182427760149, 'momentum': 0.0965182427760149, '0.2.SVM': 0.0965182427760149, 'kernel': 0.0965182427760149, 'cost': 0.0965182427760149, '9000.Logistic': 0.0965182427760149, 'regression': 0.0965182427760149, 'tested they': 0.0965182427760149, 'heuristic': 0.0965182427760149, 'three': 0.2832184100764288, 'follows': 0.19195914460735727, 'hidden': 0.14396935845551795, 'learning': 0.16357994328702444, 'number': 0.11048469689489944, 'rate': 0.5524234844744972, 'used': 0.2511015838520442, 'respectively': 0.15340980838501336, 'RBF': 0.06687663107893797, 'ANN': 0.13242892932979383, 'value': 0.09005167194425981, 'seed': 0.379743906003993, 'N': 0.126581302001331, 'F': 0.126581302001331, 'k': 0.126581302001331, 'were clustering': 0.0632906510006655, 'clusters': 0.0632906510006655, '15.C4.5': 0.0632906510006655, 'J48': 0.0632906510006655, 'confidence': 0.0632906510006655, 'factor': 0.0632906510006655, '025': 0.0632906510006655, 'folds': 0.0632906510006655, '6.CART':

(B) summary from the frequency-based approaches.

The respective ROC curve for Random Forest algorithm.

Respective F-measure and ROC area of the algorithms.

The sensitivity and specificity are statistical measurements of checkout tests.

According to Table 2, the sensitivities are less than specified for all classifiers.

According to Table 3, it is easily said all the classifiers show high specificity.

Table 2 Measures for binary classification [33].

The highest ROC curve value belongs to RF as 0.999 nearly 1.

The Table 3 displays another two-evaluation measurement.

The reported classification rate is 91.62%.

The respective values are in order 0.993, 0.985, 0.976, 0.966, 0.954, which were got by ANN,k-NN, C4.5, SVM, RBF and CART.3.3.

(C) LexRank summary

Firstly, the default values of algorithms parameters are tested and according to results of confusion matrices, it is tested to find the best accuracy that is achieved by the respective classifier.

Measure Formula Evaluation focus Accuracy TP+TN TP+TN+FP+FN Overall efficiency of a classifier Sensitivity(recall) TP TP+FN The efficiency of a classifier to categorize positively labelled data Specificity TN TN+FP Performance of a classifier when categorizes negative labels Precision TP TP+FP The data with the positive labels correctly classified by the classifier F-score 2*P*R P+R Harmonic mean between precision and recall The classifier's power to prevent misclassification Table 3 Comparison chart of the classifiers given in percentage (%).

Forest algorithm classifies CTG data with an accuracy of 99.18%.The sensitivity and the specificity values of Random Forest are 94, 18% and 99.74%, respectively.

It means that, all algorithms show a nearly same performance as a classifier, since the erro range of the classifiers is just between 0.26% and 0.67%.

The area under the ROC curve call as one of the important statistical measures, which is needed during the classification of the data.

The respective ROC curve for Random Forest algorithm.

The respective values are in order 0.993, 0.985, 0.976, 0.966, 0.954, which were got by ANN,k-NN, C4.5, SVM, RBF and CART.3.3.

The reported accuracy of the test data is 86%.In another study, it is proposed a two-step investigation of fetal heart rate data that lets for efficient prediction of the acidemiarisk.

It is reported that the overall classification accuracy is 75.61% and AUC as 0.78.

The statistical parameters such as F-score, Recall were computed to evaluate the performance and reported over-all performance are 84%, 78% and 80%, respectively.

(D) TextRank summary

CART has four parameters we have tested three of them in order number of folding for pruning as 0.2, seed as 100 and minimum number of object as 2RF has four parameters we tested three of them such as N as 13, F as 4 and T as 29.And the last one k-NN has three parameters and just two were tested k as 11 and linear search algorithm as Euclidean distance.

For the training process, it is necessary to use nine data elements while the testing process needs the residuary part to perform [39].To finish up the process, the procedure was repeated 10 times to allow each fraction of data being as a testing data.

For each algorithm, respected ROC and F-measure also computed at the end of the process. To assure for the performance of each algorithm, the value of k assigned to 10 to implement 10-fold cross validation.

The overall accuracies of the classifiers laid in the in the interval (98.42–99.13) where 99.13%, 98.96%, 98.91%, 98.74%,98.63%, 98.53% and 98.42% for the C4.5, SVM, CART, SL, ANN, RBFN and k-NN classifiers, respectively.

The area under curves were equal to 0.996, 0.993, 0.985, 0.976, 0.966, 0.966 and 0.954 for the SL, ANN, k-NN, C4.5, SVM, RBFN, and CART classifiers respectively.

This paper reveals to corroborate the investigation of the other field of application, in other words, the supremacy of Random Forest about conventional methods when handling with CTG data. Exact specification of CTG data has been critical for not just a diagnosis, but also an amendment evaluation.

However, the analysis of the sensitivities of the classifiers shows a different face of the data.

The area under the ROC curve call as one of the important statistical measures, which is needed during the classification of the data.

The performance depends on the number of accurate classifications [44] and quality index [45] specified as the geometric mean of sensitivity and specificity. According to [46] CC and QI get through to the highest value for the Weighted Fuzzy Scoring System combined with the LSVM algorithm.

Another work [47] suggest classification method using SVM but before the applying the method they eliminated the noisy data from the FHR recording and also reduced the dimension by PCA.

(E) Cluster-based summary

Sensitivity states in the rate of positive test result.

Specificity indicates to the ratio of a negative test result, which has the following formula

ROC represents the classifier performance without considering class distribution or error costs.

The area under the ROC curve call as one of the important statistical measures, which is needed during the classification of the data.

However, the analysis of the sensitivities of the classifiers shows a different face of the data.

Specificity indicates to the ratio of a negative test result, which has the following formula.

Another work [47] suggest classification method using SVM but before the applying the method they eliminated the noisy data from the FHR recording and also reduced the dimension by PCA.

This paper reveals to corroborate the investigation of the other field of application, in other words, the supremacy of Random Forest about conventional methods when handling with CTG data.

Furthermore, they proposed [50] calculation of the performance of the clustering of CTG data by k-means using the precision, recall and F-score measures.

Firstly, the default values of algorithms parameters are tested and according to results of confusion matrices, it is tested to find the best accuracy that is achieved by the respective classifier.

One of them is perpendicular to the x-axis with one unit length and similarly the other is perpendicular to the y-axis with one unit length.

Since the aim of classification is to help to clinicians during the diagnosis, 295 of the data does not help and provide useful information to apply any treatment.

(F) Reference (Human) summary

Firstly, the default values of algorithms' parameters are tested and according to results of confusion matrices, it is tested to find the best accuracy that is achieved by the respective classifier.

With the aim of calculating the performance of machine learning methods, the entire CTG data split training and testing sets, and 10-fold cross-validation, which is a famous method for evaluation, is applied afterwards.

ROC represents the classifier performance without considering class distribution or error costs

As a result of that decision trees (Random Forest is a greedy method that selects the best solution at hand when choosing classification structures that are to be applied for tests at each tree node.

The overall accuracies of the classifiers laid in the in the interval (98.42–99.13) where 99.13%, 98.96%, 98.91%, 98.74%,98.63%, 98.53% and 98.42% for the C4.5, SVM, CART, SL, ANN, RBFN and k-NN classifiers, respectively.

The area under curves were equal to 0.996, 0.993, 0.985, 0.976, 0.966, 0.966 and 0.954 for the SL, ANN, k-NN, C4.5, SVM, RBFN, and CART classifiers respectively.

The F-measures were getting the following result 0.991 for the C4.5 classifier, 0.990 for the SVM classifier and 0.989, 0987, 0986, 0985, 0.984 for the classifiers CART, SL. ANN, RBF as the last classifier k-NN, respectively.

This paper reveals to corroborate the investigation of the other field of application, in other words, the supremacy of Random Forest about conventional methods when handling with CTG data.

The sensitivity and the specificity values of Random Forest are94.18% and 99.74%, respectively.

The statistical parameters such as F-score, Recall were computed to evaluate the performance and reported over-all performance are 84%, 78% and 80%, respectively.

Appendix G: Interview Responses

Response from Interviewee A

1. There is no algorithm that works best for every problem. However, from experience, the selected models have been proven effective for such task. So, the choice of the models is right. Also, there are other models which are applicable to the same problem.
2. The problem is typically a classification problem. So, the assertion is right
3. These are good feature sets. Although, depending on how they are used, they give varying degree of performance.
4. I think the research has captured all the scenarios. The default values are designed to work on a wide variety of datasets. Using the default first and then tuning the values where needed, in respect to your dataset, to improve the performance.
5. The result (models developed) are a result of the dataset and the entire process itself but the values seem ok to me.
6. Yes, but there is room for improvement. Perhaps re-engineering the feature sets.
7. Those are the standard evaluation metrics for the machine learning to date. So, the choice is right.
8. Consider using levels of N-gram. In this work, you used the default N-gram which is 1-gram. Consider using higher grams. For example, use 2-gram and 3-gram. Observe the change of performance. It may improve or otherwise depending on your dataset.

Response from Interviewee B

1. From my experience, the selected models are good for the task. However, there are other models also which are applicable to the same problem. So, you must have justification for the settling for this 3.
2. The 'classification approach' is the right approach for this given task. Therefore, this approach is ok.
3. Yes. The feature engineering is rich enough. It captured many feature engineering scenarios
4. That is the professional thing to do. The default values work well on many datasets. However, you can twerk the values to suit your dataset.
5. The models are but the products of the dataset and how it is being used. So, models produced are only as good as your dataset. However, from the results obtained, this is good.
6. The models are good. Adding more training data may improve the models as they would learn more from the datasets.
7. The evaluation metrics are ok. They are the standard evaluation metrics available for machine learning experiment.
8. The models produce satisfactory results. However, more training (more dataset) could be considered. The models could improve.

Response from Interviewee C

1. Choosing machine learning algorithm for use depends on many factors. As contained in the thesis, the relevant factors have been clearly considered before selecting the 3 algorithms used in the training/validation. The result (prediction accuracy achieved by the models) is evident that the selected algorithms are relevant.
2. You have 6 classes for training and prediction. So, the classification approach is the most appropriate.
3. For general knowledge, these are features are ok. For a specialist field such as the scientific domains however, the TF-IFD could be reconfigured to reflect only scientific terms.
4. I am ok with the experimental settings.
5. The variation between the output from the respective model is a bit wide. Fine tuning the feature engineering could help to reduce this variation.
6. Quite effective. However, the error-rate may be high.
7. The metrics used in this research are standard evaluation metrics available. They captured every aspect of the machine learning. So, suing a selection of the metrics is the professional practice to measure the effectiveness of the models.
8. The models could be improved in 3 ways: removing some features, using different models (if not already done) and more training (dataset).